

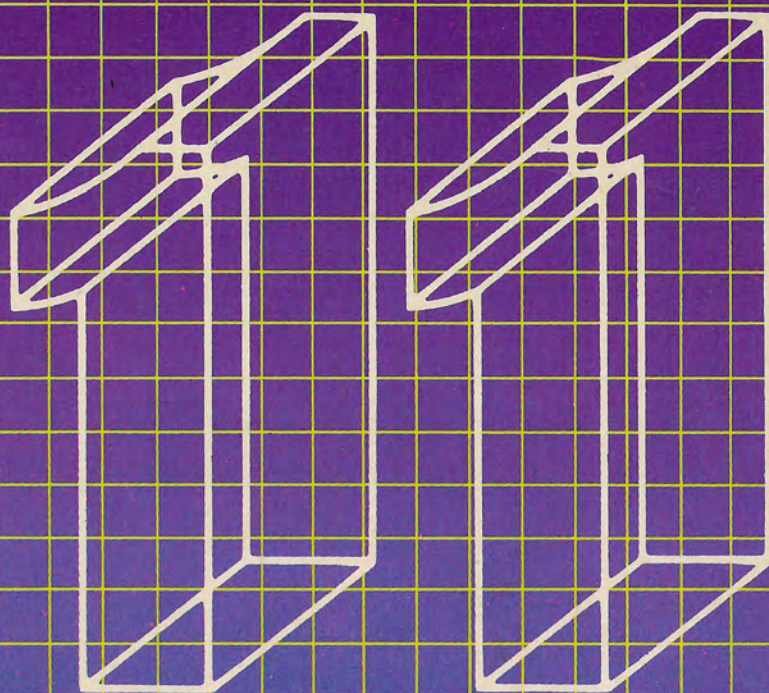
BIBLIOTECA PRACTICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

TALLER DE INFORMATICA

PROGRAMAS VALIDOS PARA AMSTRAD,

IBM, SPECTRUM, COMMODORE Y MSX



TRUCOS DE PROGRAMACION

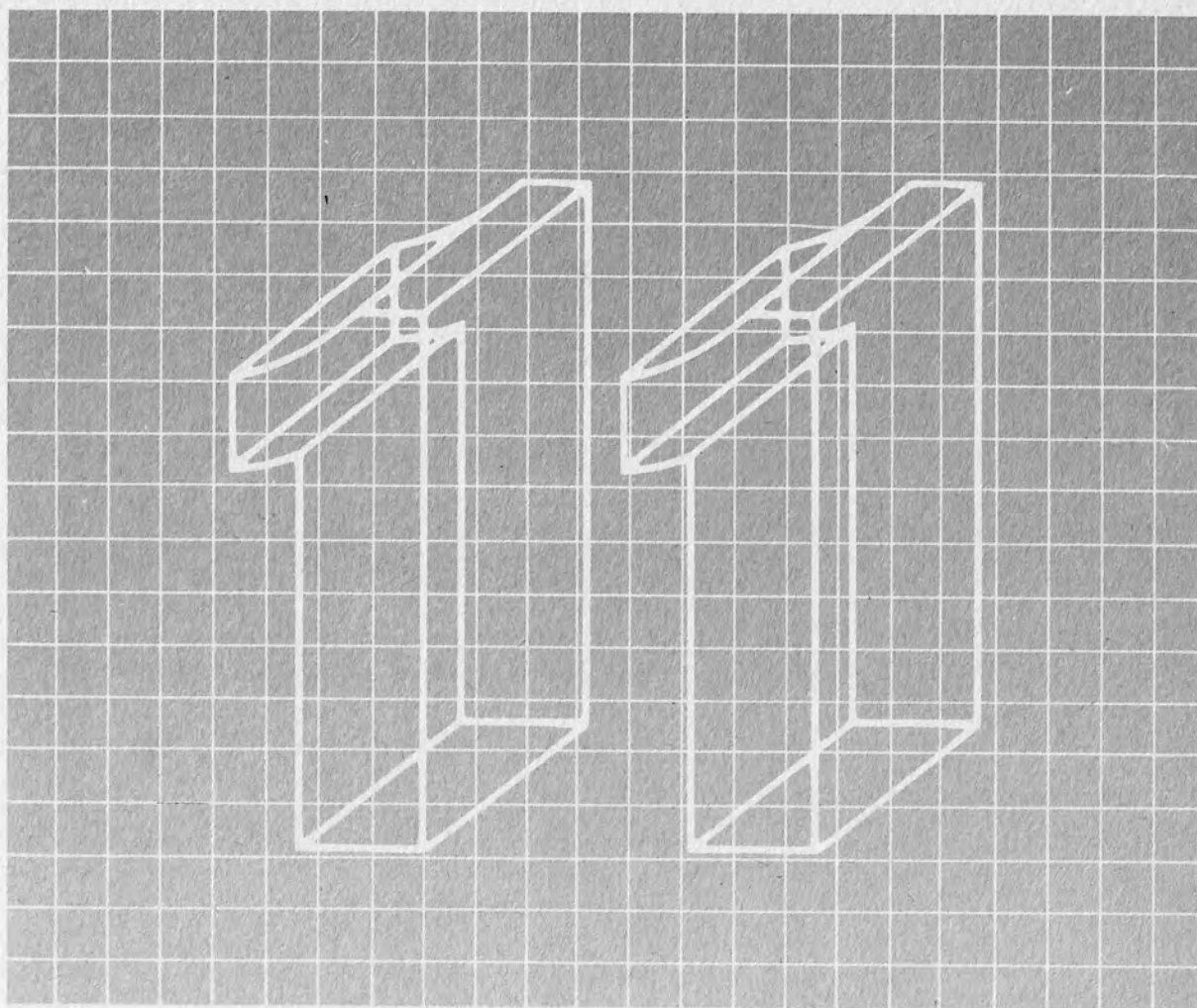
Y DE DISEÑO DE GRAFICOS

EDICIONES SIGLO CULTURAL

BIBLIOTECA PRACTICA TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD,
IBM, SPECTRUM, COMMODORE Y MSX



EDICIONES SIGLO CULTURAL

Una publicación de

EDICIONES SIGLO CULTURAL, S. A.

Director-editor:

RICARDO ESPAÑOL CRESPO.

Gerente:

ANTONIO G. CUERPO.

Directora de producción:

MARIA LUISA SUAREZ PEREZ.

Director de la colección:

JOSE ARTECHE, Ingeniero de Telecomunicación

Diseño:

BRAVO-LOFISH.

Maquetación:

D. S. M.

Dibujos:

JOSE OCHOA

Tomo 11. «Experiencias y prácticas en logo», Carlos Albert, Técnico de Informática; Adoración Llena Jubero, Técnico de Informática. «Manejo de sprites y elementos gráficos», «Trucos y rutinas básicas», Francisco Morales. Técnico de Informática. «Aprende con el ordenador», AULA DE INFORMÁTICA APLICADA: Soledad Tamariz-Martel, Diplomada en Telecomunicación; Francisco Blanca, Diplomado en Telecomunicación; Alejandro Marcos, Licenciado en Química; Fernando Suero, Diplomado en Telecomunicación. «El Taller del Hardware», Carlos Rey, Ingeniero Industrial. «Pequeña historia de la Informática», «Temas monográficos de vanguardia», «Terminología», «Vocabulario de Informática», Blanca Arbizu, Técnico de Informática.

Ediciones Siglo Cultural, S. A.

Dirección, redacción y administración:

Pedro Teixeira, 8-2.º. Teléf. 810 52 13. 28020 Madrid.

Publicidad:

Gofar Publicidad, S. A. Benito de Castro, 12 bis. 28028 Madrid.

Distribución en España:

COEDIS, S. A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.

Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:

CADE, S.R.L. Pasaje Sud América, 1532. Teléf.: 21 24 64.

Buenos Aires - 1.290. Argentina.

Todos los derechos reservados. Este libro no puede ser, en parte o totalmente, reproducido, memorizado en sistemas de archivo, o transmitido en cualquier forma o medio, electrónico, mecánico, fotocopia o cualquier otro, sin la previa autorización del editor.

ISBN del tomo: 84-7688-076-6

ISBN de la obra: 84-7688-047-2

Fotocomposición:

ARTECOMP, S. A. Albarracín, 50. 28037 Madrid.

Imprime:

MATEU CROMO. PINTO (Madrid).

© Ediciones Siglo Cultural, S. A., 1986

Depósito legal: M-43.185-1986

Printed in Spain - Impreso en España.

Suscripciones y números atrasados:

Ediciones Siglo Cultural, S. A.

Pedro Teixeira, 8-2.º. Teléf. 810 52 13. 28020 Madrid.

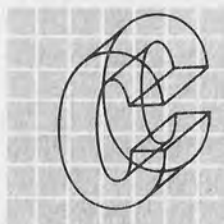
Abril, 1987.

P.V.P. Canarias: 355.—Ptas.

INDICE

■ EXPERIENCIA Y PRACTICAS EN LOGO	5
■ MANEJO DE SPRITES Y ELEMENTOS GRAFICOS	15
■ TRUCOS Y RUTINAS BASICAS	23
■ EL TALLER DEL HARDWARE	37
■ APRENDER CON EL ORDENADOR	43
■ PEQUEÑA HISTORIA DE LA INFORMATICA	55
■ TEMAS MONOGRAFICOS DE VANGUARDIA	57
■ VOCABULARIO DE INFORMATICA	59

EXPERIENCIA Y PRACTICAS EN LOGO



CUANDO definimos los procedimientos con entrada, utilizamos una variable mediante la cual introducimos un valor en el procedimiento para que sea utilizado en todas aquellas órdenes que contengan el nombre de la

variable.

Para utilizar esta variable lo único que tenemos que hacer es indicar al lado del nombre del procedimiento, el nombre con el que queremos identificarla precedido de dos puntos. En la orden en donde vayamos a usar la variable tenemos que colocar también el nombre de ésta.

A la hora de ejecutar un procedimiento que contenga una variable, tenemos que indicar a continuación del nombre del procedimiento el valor que asignamos a dicha variable.

— Para definir un procedimiento con entrada:

— Para nombre de procedimiento:
nombre de variable.

— Para ejecutarlo:
nombre de procedimiento valor de la variable.

Vamos a conocer mejor lo que es una variable de este tipo, cómo funciona y las posibilidades que nos ofrecen.

Una variable (en general), es una pequeña zona de memoria que está identificada por un nombre y que no tiene un valor fijo o determinado. Esta zona la utilizamos para

guardar en ella valores que dependiendo de lo que necesitemos serán unos u otros. Un valor asignado a una variable permanecerá en ella hasta que volvamos a cambiarlo. A esta zona podemos acceder tanto para guardar algo en ella como para, simplemente, consultar su contenido.

Una constante es un valor fijo, por ejemplo, el valor 1.

$A = 5$; asignamos una constante a una variable.

$A = B$; asignamos el valor de una variable a otra variable.

Este tipo de variables que utilizamos en los procedimientos tienen una diferencia con otros tipos de variables.

Con el siguiente ejemplo podemos verlo.

Define el siguiente procedimiento:

```
? PARA EJEMPLO :DATO
> ESCRIBE :DATO
> REPITE 3 [AV :DATO GD 120 ]
> ESCRIBE :DATO
> FIN
```

Es un procedimiento que utiliza una variable (:DATO) en tres sitios diferentes.

Primero ordenamos que escriba el valor de la variable, luego que dibuje un triángulo en el que sus lados miden el valor que le demos a :DATO, y, por último, que vuelva a escribirlo.

Si ahora lo ejecutamos:

```
? EJEMPLO 20
```

Escritura infija de una expresión: los operadores se colocan entre los operandos.

EXPERIENCIA Y PRACTICAS EN LOGO

En la pantalla obtenemos y en este orden lo siguiente:

- En la zona texto, un 20.
- En la zona de gráficos, el dibujo de un triángulo.
- En la zona de texto, otro 20.

A la hora de definir el procedimiento, una zona de la memoria se ha reservado y se le ha dado el nombre de DATO. Cuando al ejecutarlo introducimos un valor (20), éste pasa a esa zona de la memoria. Empiezan a realizarse las órdenes y cuando encuentra una que necesita el valor de la variable DATO, va a la memoria, consulta en la zona que tiene destinada para almacenar variables y busca una que se llame DATO. Una vez que la ha encontrado, saca el valor que contiene y lo deposita en el lugar donde se requiera.

Continúa haciendo las órdenes y en todas las que encuentre el nombre de la variable, realiza el mismo proceso.

Una vez que se haya ejecutado el procedimiento introduce la siguiente orden:

```
? ESCRIBE :DATO
```

Volvemos a solicitar, al igual que hacíamos dentro del procedimiento, que se escriba en la pantalla el valor que contiene la variable DATO.

Como resultado obtenemos el siguiente mensaje:

```
DATO NO TIENE VALOR
```

¿Qué ha ocurrido? ¿Por qué ahora no obtenemos el valor 20?

La variable se ha borrado y con ella su contenido. Sólo ha permanecido durante el tiempo que ha durado la ejecución del procedimiento.

Este tipo de variables reciben el nombre de:

VARIABLES LOCALES

Vamos a ver algunas de las formas con las que podemos asignar valores a las variables.

- Indicando un valor constante:

```
? EJEMPLO 3
```

- Indicando una expresión aritmética:

```
? EJEMPLO (4 * 5) / 2
```

- Indicando el nombre de otra variable:

Supongamos que tenemos dos procedimientos y que cada uno de ellos utiliza una variable diferente:

```
? PARA A :DATO 1
```

```
> AV :DATO 1
```

```
> GD 90
```

```
> B :DATO 1
```

```
> FIN
```

```
? PARA B :DATO 2
```

```
> BP
```

```
> REPITE 4 [AV :DATO 2 GD 90]
```

```
> FIN
```

En el procedimiento A se hace una llamada al procedimiento B que necesita un valor para la variable DATO 2. El valor que pasa al procedimiento B es el mismo que se ha utilizado en el A.

- Indicando el nombre de una variable en combinación con un valor constante:

En el ejemplo anterior se podría realizar lo siguiente:

```
? PARA A :DATO 1
```

```
> AV :DATO 1
```

```
> GD 90
```

```
> B :DATO 1 + 10
```

```
> FIN
```

El valor que pasa a la variable utilizada en el procedimiento B es el mismo que se utiliza en A, pero aumentado en 10 unidades.

■ Procedimientos con dos entradas

Si en los procedimientos con una entrada utilizamos una variable, es lógico pensar que en los de dos entradas se van a utilizar dos.

A la hora de definir el procedimiento tenemos que especificar un nombre para cada una de las variables que vamos a utilizar:

```
? PARA EJEMPLO :DATO 1 :DATO 2
```

Existe un orden de prioridad para evaluar una expresión.

Tienen que ir separadas por un espacio y precedidas de dos puntos.

Una vez definidas las variables que se van a utilizar, podemos hacer uso de ellas en cualquier parte del procedimiento:

```
> REPITE 4 [AV :DATO 1 GD 90]
> AV :DATO 2
> FIN
```

A la hora de ejecutarlo tenemos que especificar un valor para cada una de las variables que deberán ir acompañando al nombre del procedimiento y separadas por un espacio:

```
? EJEMPLO valor de la 1.ª variable valor de la 2.ª variable.
```

Los valores se asignarán a las variables según el orden en el que se hayan escrito, es decir, el primer valor que aparezca a continuación del nombre del procedimiento, se asigna a la primera variable que se haya definido.

Estas variables, al igual que en el caso de los procedimientos de una entrada, sólo tendrán valor el tiempo que dure la ejecución del procedimiento.

Una vez que sabemos cómo se definen los procedimientos con dos entradas, vamos a ver su utilidad y aplicaciones.

Si defines el siguiente procedimiento:

```
? PARA RECTANGULO
> REPITE 2 [AV 50 GD 90 AV 25 GD 90]
> FIN
```

y ahora lo ejecutas:

```
? RECTANGULO
```

Obtendrías en la pantalla el siguiente dibujo:

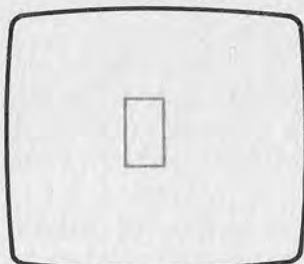


Fig. 1.

El rectángulo que has dibujado tiene unas dimensiones determinadas. Una altura de 50 pasos o puntos y una anchura de 25.

Estos dos valores son los que debemos ir variando si queremos dibujar rectángulos de diferentes tamaños.

Bastaría con utilizar dos variables en sustitución de las dos constantes que aparecen en el procedimiento. Una que determine la altura y otra la anchura.

```
? PARA RECTANGULO :ALTO :ANCHO
```

```
> REPITE 2 [AV :ALTO GD 90 AV :ANCHO GD 90]
> FIN
```

El mismo rectángulo de antes podemos ahora obtenerlo de la siguiente forma:

```
? RECTANGULO 50 25
```

Ahora, con sólo ir pasando distintos valores al procedimiento, podemos dibujar rectángulos de todos los tamaños

Compruébalo con las siguientes órdenes:

```
? RECTANGULO 20 10
```

```
? RECTANGULO 40 20
```

```
? RECTANGULO 10 20
```

```
? RECTANGULO 20 40
```

En la pantalla te aparecerá lo siguiente:

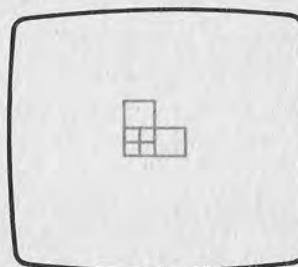


Fig. 2.

Utilizando el procedimiento rectángulo podemos también hacer bonitos dibujos:

```
? REPITE 30 [RECTANGULO 50 5 GD 12]
```

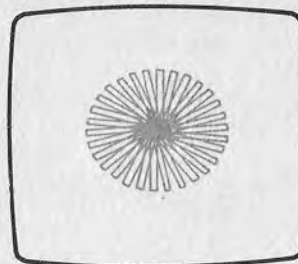


Fig. 3.

Hay órdenes que no admiten variables.

EXPERIENCIA Y PRACTICAS EN LOGO

36] ? REPITE 10 [RECTANGULO 50 30 GD

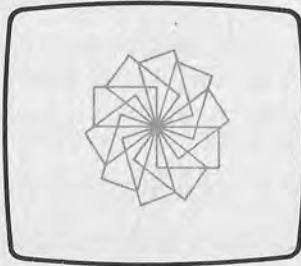


Fig. 4.

Con el siguiente ejemplo demostramos cómo los procedimientos con dos entradas pueden ser de gran utilidad a la hora de dibujar algo donde queremos tener opción a elegir por ejemplo, tamaños o números de veces que queremos que aparezca un dibujo determinado.

Definimos un primer procedimiento que llamamos BARRIO, que utiliza dos variables, CASAS y PISOS. La primera determina el número de casas que queremos dibujar y la segunda la altura de éstas. En este procedimiento vamos determinando la posición en donde tiene que ir cada casa y llamando al procedimiento ALTURA:

```
? PARA BARRIO :CASAS :PISOS
> BP
> SL
> PONPOS [-100 -50]
> REPITE :CASAS [ALTURA SL RE :PISOS * 20 GD 90 AV 70 GI 90]
> FIN
```

El procedimiento ALTURA dibuja, en primer lugar, la estructura de la casa. La altura de la casa viene determinada por el número de pisos que tiene multiplicado por 20, que es la altura que hemos fijado para cada uno.

A continuación hay una llamada al procedimiento PUERTA, que es el que dibuja la puerta de entrada al edificio. Y por último, llamamos al procedimiento VENTANAS tantas veces como pisos hay. Este dibuja dos ventanas y un balcón.

```
? PARA ALTURA
> BL
> REPITE 2 [AV (:PISOS + 1) * 20 GD 90 AV 60 GD 90]
```

```
> PUERTA
> REPITE :PISOS [VENTANAS SL AV 20 GI 90 AV 45 GD 90 RE 25]
> FIN
```

Este es el procedimiento para dibujar la puerta:

```
? PARA PUERTA
> GD 90 AV 20 GI 90
> REPITE 2 [AV 15 GD 90 AV 20 GD 90]
> GD 90 AV 10 GI 90
> AV 15 RE 15 GI 90
> AV 30 GD 90
> FIN
```

Y éste el que dibuja las ventanas y el balcón:

```
? PARA VENTANAS
> SL
> AV 25 GD 90
> AV 7 GI 90 BL
> REPITE 4 [AV 8 GD 90]
> SL RE 5 GI 90
> RE 17 GD 90 BL
> REPITE 2 [AV 5 GD 90 AV 12 GD 90]
> REPITE 6 [GD 90 AV 2 GI 90 AV 5 RE 5]
> AV 5 GI 90 AV 2 GD 90
> REPITE 3 [AV 8 GI 90]
> SL AV 19 GI 90
> BL
> REPITE 4 [AV 8 GD 90]
> FIN
```

Con todos estos procedimientos podemos dibujar tanto el número de casas como la altura que queramos, pero si elegimos valores superiores a tres para el número de casas y de 6 para el de pisos, las dimensiones de la pantalla no nos permite verlo correctamente, ya que se sobrepasan.

A la hora de ejecutar estos procedimientos, basta con introducir el nombre del procedimiento general (BARRIO) y a continuación indicar el valor de las dos variables que se utilizan.

Si queremos que aparezca una sola casa de cinco pisos:

```
? BARRIO 1 5
```

Dos formas para expresar las operaciones aritméticas: escritura infija y escritura prefija.

Y obtendríamos el siguiente dibujo:

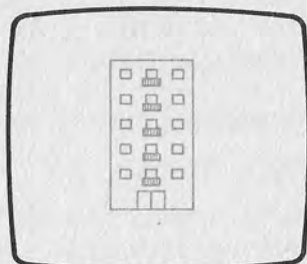


Fig. 5.

Para tres casas de cuatro pisos:

? BARRIO 3 4

Y obtendríamos este otro dibujo

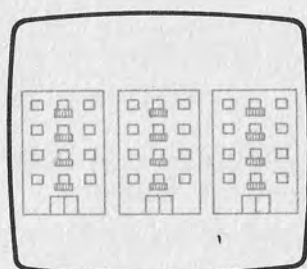


Fig. 6.

Observa cómo con un solo procedimiento de dos entradas conseguimos dibujar cualquier polígono regular de cualquier tamaño.

```
? PARA FIGURA :N :L
> REPITE :N [AV :L GD 360 / :N]
> FIN
```

:N determina el número de lados del polígono
y :L la longitud de cada uno de los lados.

Podemos ejecutarlo directamente

? BP GI 90 FIGURA 6 20

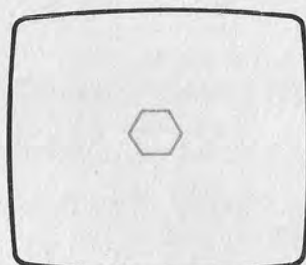


Fig. 7.

? BP GI 90 FIGURA 8 10

o también podemos definir un procedimiento que realice las figuras que queramos.

```
? PARA CONJUNTO
```

```
> OT
> BP
> GI 90
> FIGURA 6 20
> ESPERA 30
> BP
> FIGURA 8 10
> ESPERA 30
> BP
> FIGURA 9 10
> FIN
```

Para ejecutar este procedimiento te-
clea:

```
? CONJUNTO
```

Con CONJUNTO dibujamos tres figuras geométricas diferentes. Al terminar de dibujar una figura y tras una breve espera, la pantalla se borra y se ejecuta la siguiente figura.

Otro ejemplo de procedimiento con dos entradas en este dibujo de un fichero. Podemos elegir el número de cajones y el tamaño de cada uno de ellos.

FICHERO es el procedimiento de dos entradas.

:V para el número de cajones.
:H para la altura de cada cajón.

```
? PARA FICHERO :V :H
> BP
> SL
> PONPOS [-15 -70]
> REPITE :V [CAJON :H]
> PONPOS [-20 -75]
> BL
> REPITE 2 [AV 10 + :V * H GD 90 AV
40 GD 90]
> FIN
```

El dibujo de los cajones lo realizamos con el procedimiento de una entrada CAJON.

```
? PARA CAJON :H
> BL
> REPITE 2 [AV :H GD 90 AV 30 GD 90]
> SL
> PONX -5
> PONY COORY + (:H / 3)
> BL
```

**En la escritura prefija el signo de la operación
va delante de los operandos.**

EXPERIENCIA Y PRACTICAS EN LOGO

```
> REPITE 2 [AV 3 GD 90 AV 10 GD 90]
> SL
> PONY COORY + (2 * (:H / 3))
> PONX -15
> FIN
```

Para ejecutar el dibujo llamamos al procedimiento FICHERO, indicando dos valores, el primero para el número de cajones y el segundo para la altura de cada cajón.

```
? FICHERO 10 10
```

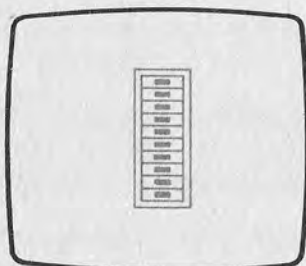


Fig. 8.

```
? FICHERO 4 30
```

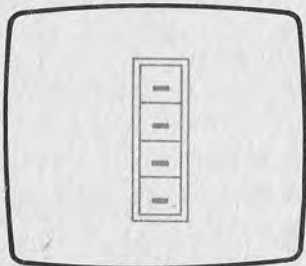


Fig. 9.

■ Escritura prefija de las operaciones aritméticas

Ya conocemos la sintaxis y la forma de la escritura infija (los operadores se colocan entre los operandos); vamos a ver ahora la escritura prefija de algunas operaciones aritméticas.

```
? ESCRIBE SUMA 5 6
```

Recordamos que ESCRIBE es una primitiva que indica que el resultado de la operación, en este caso una suma, debe aparecer en la pantalla.

Con la orden anterior obtenemos el resultado de 5 más 6.

En la escritura prefija el tipo de operación precede a los operandos y sólo admite a dos de ellos, que pueden ser números o expresiones cuyo resultado sea numérico.

```
? ESCRIBE SUMA n1 n2
```

Observa la equivalencia entre la escritura infija y la escritura prefija de algunas operaciones aritméticas con estos ejemplos.

```
? ESCRIBE 5 -3  equivale a ? ESCRIBE RESTA 5 3
? ESCRIBE 5 * 3  equivale a ? ESCRIBE PRODUCTO 5 3
? ESCRIBE 15 / 3 equivale a ? ESCRIBE COCIENTE 5 3
? ESCRIBE 5 +     equivale a ? ESCRIBE SUMA 5 COCIENTE 10 5
```

En las divisiones, si el divisor es cero, el LOGO muestra el mensaje:

IMPOSIBLE DIVIDIR POR CERO

Aunque es posible mezclar las dos formas de escritura en una misma expresión aritmética, es conveniente no hacerlo.

Las operaciones prefijas se evalúan según su orden de escritura; así:

```
? ESCRIBE SUMA 4 3 RESTA 5 PRODUCTO 4
8
```

suma 4 y 3, al resultado le restas 5 y lo multiplicas por 4. Equivale a:

```
? ESCRIBE (4 + 3 - 5) * 4
8
```

Hemos visto cuatro operaciones en forma prefija.

SUMA n1 n2	suma n1 y n2
RESTA	resta n2 de n1
PRODUCTO	multiplica n1 y n2
COCIENTE	divide n1 entre n2

La operación COCIENTE sólo devuelve la parte entera de la división.

Nota: Algunas versiones de LOGO no disponen de la operación resta en escritura prefija.

La escritura prefija necesita para operar dos valores: ni más ni menos.

Practica esta nueva forma de escritura con los siguientes ejemplos:

- ? ESCRIBE SUMA 5 6 7
11
NO SE QUE DEBO HACER CON 7
- ? ESCRIBE SUMA 20 2 PRODUCTO
10
22
NO SE QUE DEBO HACER CON 10
- ? ESCRIBE SUMA 20 PRODUCTO
10 COCIENTE 20 2
120

El segundo dato de la suma es el resultado de la multiplicación de 10 con el resultado de la división de 20 entre 2. Otra forma más clara de escribir esta misma expresión sería utilizando los paréntesis.

ESCRIBE SUMA 20 (PRODUCTO 10 (COCIENTE 20 2))

- ? ESCRIBE SUMA 40 50 + 20
110
- ? ESCRIBE SUMA 500
500
- ? ESCRIBE PRODUCTO 500
500
- ? ESCRIBE COCIENTE 500
FALTAN DATOS PARA COCIENTE
- ? ESCRIBE SUMA COCIENTE 10 2 5
10
- ? ESCRIBE SUMA 5 COCIENTE 10 2
10
- ? ESCRIBE COCIENTE 5 3
1

■ Gestiones con ficheros LOGO

Al guardar uno o varios procedimientos en el disco, el LOGO crea un fichero con el nombre que hallamos determinado y por defecto .LOG, en este fichero el/los procedimientos se guardan codificados en ASCII.

Que los ficheros creados por el LOGO estén en ASCII, nos permiten acceder a ellos

con cualquier comando o programa que gestione ficheros ASCII.

Por ejemplo, desde el sistema operativo (DOS) si tecleamos:

TYPE nombre.LOG

nos aparecerá por pantalla el listado de todos los procedimientos que contenga el fichero nombre.LOG.

Programas que permitan gestionar estos ficheros son, por ejemplo, los editores. Desde los que podemos acceder a cualquier fichero creado por LOGO.

EDLIN nombre.LOG

EDLIN es un editor de líneas con el que podemos manipular cualquier fichero codificado en ASCII.

Si una vez cargado el fichero .LOG en el editor modificamos, introducimos o borramos alguna orden y luego guardamos el fichero con el mismo nombre, al volver a cargarlo con el LOGO observaremos que todo lo que hemos hechos en el editor aparece en los procedimientos.

■ Ejercicios

1. Utilizando un procedimiento de dos entradas y dos de una entrada, intenta realizar el siguiente dibujo, teniendo en cuenta que se debe poder aumentar o disminuir el tamaño del círculo sin que varíe el del triángulo, y viceversa.

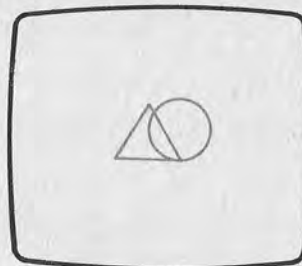


Fig. 10.

2. Define los procedimientos necesarios para dibujar una puerta cuyo tamaño puedas variar sin modificar los procedimientos.
3. Realiza el siguiente dibujo, pudiendo indicar al ejecutarlo la posición en X y el color.

Es conveniente no mezclar en una misma expresión aritmética los dos tipos de escritura.

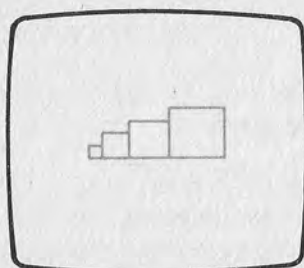


Fig. 11.

4. Dibuja esta cara en diferentes posiciones sin que para ello tengas que modificar los procedimientos.

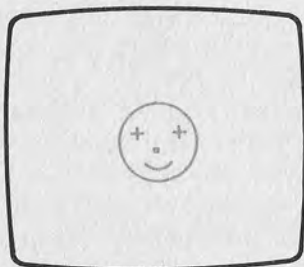


Fig. 12

5. ¿Son correctas las órdenes siguientes?
- ? ESCRIBE COCIENTE 400 PRODUCTO 40
 - ? ESCRIBE PRODUCTO 400 SUMA 40 -10
 - ? ESCRIBE COCIENTE 400 SUMA 40 -10
 - ? PARA EJEMPLO :A B
 - ? AV :LADO + 10
 - ? REPITE VECES [AV :VECES * 5 GD 90]

■ Soluciones a los ejercicios

1:

Este primer procedimiento tiene dos entradas, una la utilizamos como parámetro del procedimiento CIRCULO y la otra como parámetro del procedimiento TRIANGULO.

```
? PARA DIBUJO :C :T
> BP
```

```
> GI 90
> CIRCULO :C
> TRIANGULO :T
> FIN
```

Procedimiento para dibujar un triángulo cuyo lado depende del valor que especifiquemos cuando llamamos a TRIANGULO.

```
? PARA TRIANGULO :T
> REPITE 3 [AV :T GD 120]
> FIN
```

Procedimiento para dibujar un círculo, el tamaño lo determinará el valor de :C.

```
? PARA CIRCULO :C
> REPITE 36 [AV :C GD 10]
> FIN
```

Al ejecutar el dibujo debes especificar dos datos, el primero para determinar el tamaño del círculo y el segundo para determinar el tamaño del triángulo.

```
? DIBUJO 3 30
```

2:

El procedimiento DIBUJO inicializa la pantalla y determina el orden de ejecución del dibujo. Es un procedimiento de dos entradas, la altura y la base.

La puerta la dibujamos utilizando rectángulos, para ello desde DIBUJO se realizan varias llamadas al procedimiento RECTAN.

```
? PARA DIBUJO :H :B
> BP PONCL 2
> SL
> PONY -( :H / 2)
> BL
> RECTAN :H :B
> SL GI 90 AV :B
> PONRUMBO 0
> BL
> RECTAN :H :B
> SL
> PONX :B / 4
> PONY 0
> BL
> RECTAN (:H / 3 - 3) (:B / 2)
> SL
> PONY COORY - 1
```

Los ficheros que crea el LOGO cuando guardamos en el disco procedimientos, están codificados en ASCII.

```

> PONCL 2
> BL
> RELLENA
> PONCL 2
> SL PONX -( :B - :B / 4)
> PONY 0
> BL
> RECTAN (:H / 3 - 3) (:B / 2)
> SL
> PONY COORY -3
> PONCL 1
> BL
> RELLENA
> OT
> FIN

```

RECTAN dibuja un rectángulo de base :B y altura :H, datos que se determinan al ejecutar el procedimiento DIBUJO.

```

? PARA RECTAN :H :B
> REPITE 2 [ AV :H GD 90 AV :B GD 90 ]
> FIN

```

Ejecución del dibujo

```

? DIBUJO 60 30

```

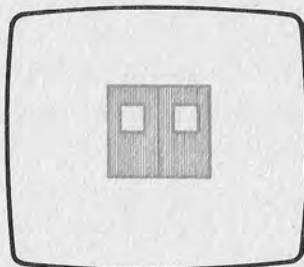


Fig. 13.

3:

Definimos el procedimiento DIBUJO con dos entradas, una para determinar el color y la otra para indicar la coordenada X, en la que situaremos el dibujo.

```

? PARA DIBUJO :X :C
> BP
> SL
> CENTRO
> PONX :X
> PONCL :C
> BL
> CUADRADO 10
> CUADRADO 20
> CUADRADO 30

```

```

> CUADRADO 40
> FIN

```

En DIBUJO inicializamos la pantalla, posicionamos a la Tortuga, inicializamos el color y dibujamos cuatro cuadrados de diferentes tamaños; para ello, llamamos al procedimiento CUADRADO indicando la longitud de lado, ya que la entrada de este procedimiento determina el tamaño del cuadrado.

```

? PARA CUADRADO :L
> REPITE 4 [ AV :L GD 90 ]
> GD 90 AV :L
> GI 90
> FIN

```

Lo ejecutamos, por ejemplo:

```

? DIBUJO -40 2

```

El dibujo lo posicionamos en la coordenada X = - 40.

El color de las figuras será aquél cuyo código sea igual a 2.

4:

Las entradas del procedimiento CARA son las que fijan la posición del dibujo.

```

:X para la coordenada X
:Y para la coordenada Y

```

```

? PARA CARA :X :Y
> BP
> SL
> PONY :Y PONX :X
> GI 90
> BL
> REPITE 4 [ AV 3 GD 90 ]
> SL
> PONRUMBO 0
> PONY :Y + 15
> GI 90
> PONX :X - 10
> OJO
> PONRUMBO 0
> SL
> PONY :Y + 15
> PONX :X + 20
> GI 90
> OJO
> SL
> PONY :Y PONX :X
> PONRUMBO 0
> RE 10 GI 90 AV 10 GI 135 BL

```

Una constante es un dato fijo.

EXPERIENCIA Y PRACTICAS EN LOGO

```
> REPITE 10 [ AV 2 GI 10 ]
> SL
> PONY :Y
> PONX :X
> PONRUMBO 0
> RE 22
> GI 90 RE 2 BL
> REPITE 36 [ AV 5 GD 10 ]
> FIN
```

En CARA hacemos dos llamadas a otro procedimiento para dibujar los ojos.

```
? PARA OJO
> BL
> AV 10 SL GI 180 AV 5 GI 90
> BL AV 5 RE 10
> FIN
```

Al ejecutar el procedimiento CARA debes especificar dos valores: el primero para la coordenada X y el segundo para la coordenada Y.

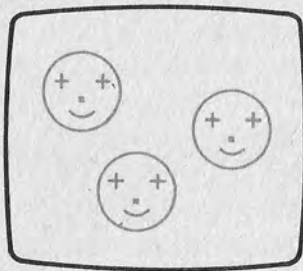


Fig. 14.

Dibujemos tres caras en posiciones diferentes:

```
? BP
? CARA -40 30
? CARA 0 -30
? CARA 30 0
```

5:

```
— ? ESCRIBE COCIENTE 400 PRO-
DUCTO 40
CORRECTO
```

El resultado es 10, la división de 400 entre 40.

```
— ? ESCRIBE PRODUCTO 400 SUMA
40 -10
CORRECTO
```

El resultado es 12000, corresponde al producto de 400 con el resultado de $40 + (-10)$.

```
— ? ESCRIBE COCIENTE 400 SUMA
-40
INCORRECTO
```

El LOGO muestra en pantalla el mensaje IMPOSIBLE DIVIDIR POR CERO, ya que $40 + (-40)$ es cero.

```
— ? PARA EJEMPLO :A B
INCORRECTO
```

La variable B debe ir precedida de dos puntos.

```
— ? AV :LADO + 10
CORRECTO
```

El resultado final de lo que se avanza es 10 más el valor de la variable LADO.

```
— ? REPITE VECES [ AV :VECES * 5
GD 90 ]
INCORRECTO
```

Hay que indicar la variable VECES como :VECES

A una variable se le puede asignar un valor que venga determinado por una expresión aritmética.

Al ejecutar un procedimiento de dos entradas tenemos que detallar, junto al nombre del procedimiento, dos datos.

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS



CONTINUANDO con el control de movimiento de figuras mediante el teclado, en este nuevo tomo vamos a ver algunas cosas más sobre cómo controlar una o más figuras que se muevan por la pantalla.

Ya vimos cómo hacer que ciertas teclas del teclado nos sirviesen para movernos por la pantalla. Hablamos de que la elección de dichas teclas es muy importante de cara al futuro usuario del programa que estamos realizando. Dijimos que era muy importante elegir una secuencia de teclas que sean cómodas al usuario, aunque su programación resulte más complicada.

El programa que vamos a ver a continuación nos servirá para que el usuario redefina sus propias teclas de control. Antes de nada veamos dicho programa.

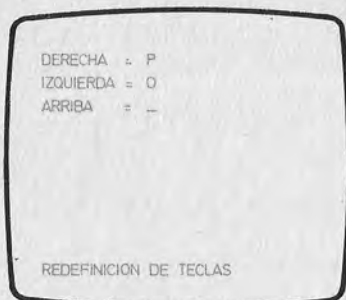


Fig. 1. Si nuestros programas permiten redefinir las teclas, éstos serán mucho más fáciles al usuario.

```
1000 REM *****
1010 REM * MOVIMIENTO CONTROLADO DE UN *
1020 REM * ASTERISCO. *
1030 REM * *
1040 REM * LAS TECLAS DE MOVIMIENTO *
1050 REM * SON REDEFINIBLES. *
1060 REM *****
```

```
1070 REM
1080 REM *** DEFINICION DE LAS TECLAS ***
1090 REM
1100 CLS
1110 PRINT "DEFINICION DE LAS TECLAS"
1120 PRINT "-----"
1130 PRINT
1140 PRINT " Cada vez que aparezca una"
1150 PRINT "direccion en la pantalla ,"
1160 PRINT "pula la tecla que quieres"
1170 PRINT "que controle dicha direc. "
1180 PRINT:PRINT:PRINT
1190 DIM B$(8)
1200 FOR I=1 TO 8
1210   READ A$
1220   LOCATE 15,1
1230   PRINT " ":REM 25 ESPACIOS
1240   LOCATE 15,1
1250   PRINT A$;
1260   A$=INKEY$:IF A$="" THEN GOTO 1260
1270   LET B$(I)=A$
1280   PRINT " = ";A$
1290   FOR J=1 TO 200
1300     NEXT J
1310 NEXT I
1320 REM
1330 REM *** VISUALIZACION DEL LAS TECLAS ***
1340 REM
1350 CLS
1360 PRINT "TECLAS DEFINIDAS"
1370 PRINT "=====
1380 PRINT:PRINT
1390 PRINT "Las teclas definidas son:"
1400 PRINT
1410 RESTORE 1550
1420 FOR I=1 TO 8
1430   LOCATE 7+I,1
1440   READ A$
1450   PRINT B$(I);" = ";A$
1460 NEXT I
1470 PRINT:PRINT
1480 PRINT "ESTAS DE ACUERDO (S/N)"
1490 LET A$=INKEY$:IF A$="" THEN GOTO 1490
1500 IF A$="N" OR A$="n" THEN RUN
1510 IF A$<>"S" AND A$<>"s" THEN GOTO 1490
1520 REM
1530 REM *** LINEAS DE DATA ***
1540 REM
1550 DATA "IZQUIERDA", "DERECHA"
```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

```

1560 DATA "ARRIBA", "ABAJO"
1570 DATA "ARRIBA-IZQ", "ARRIBA-DER"
1580 DATA "ABAJO-IZQ", "ABAJO-DER"
1590 REM
1600 REM *** PROGRAMA GENERAL ***
1610 REM
1620 CLS
1630 LET X1=10
1640 LET Y1=10
1650 LET X=10
1660 LET Y=10
1670 LOCATE Y1,X1
1680 PRINT " "
1690 LOCATE Y,X
1700 PRINT "*";
1710 LET SW=0:LET A$=INKEY$
1720 FOR I=1 TO 8
1730 IF A$=B$(I) THEN LET SW=1
1740 NEXT I
1750 IF SW=0 THEN GOTO 1710
1760 LET X1=X
1770 LET Y1=Y
1780 REM
1790 REM *** ELECCION DEL MOVIMIENTO ***
1800 REM
1810 IF A$=B$(2) THEN LET X=X+1
1820 IF A$=B$(1) THEN LET X=X-1
1830 IF A$=B$(3) THEN LET Y=Y-1
1840 IF A$=B$(4) THEN LET Y=Y+1
1850 IF A$=B$(5) THEN LET X=X-1:LET Y=Y-1
1860 IF A$=B$(6) THEN LET X=X+1:LET Y=Y-1
1870 IF A$=B$(8) THEN LET X=X+1:LET Y=Y+1
1880 IF A$=B$(7) THEN LET X=X-1:LET Y=Y+1
1890 REM
1900 REM
1910 IF X>40 THEN LET X=40
1920 IF X<1 THEN LET X=1
1930 IF Y>24 THEN LET Y=24
1940 IF Y<1 THEN LET Y=1
1950 GOTO 1670

```

Las modificaciones que hay que realizar en el programa para ordenadores distintos del IBM y compatibles son las siguientes:

COMMODORE:

```

1100 PRINT CHR$(147)
1220 POKE 214,15:POKE 211,0
1240 POKE 214,15:POKE 211,0
1260 GET A$:IF A$="" THEN GOTO 1260
1350 PRINT CHR$(147)
1430 POKE 214,7+I:POKE 211,0
1490 GET A$:IF A$="" THEN GOTO 1490
1620 PRINT CHR$(147)
1670 POKE 214,Y1:POKE 211,X1
1690 POKE 214,Y:POKE 211,X
1910 IF X>29 THEN LET X=29
1920 IF X<0 THEN LET X=0
1930 IF Y>23 THEN LET Y=23
1940 IF Y<0 THEN LET Y=0

```

MSX:

```

1220 LOCATE 1,15
1240 LOCATE 1,15
1430 LOCATE 1,7+I
1670 LOCATE X1,Y1
1690 LOCATE X,Y

```

AMSTRAD:

```

1220 LOCATE 1,15
1240 LOCATE 1,15
1430 LOCATE 1,7+I
1670 LOCATE X1,Y1
1690 LOCATE X,Y
1930 IF Y>25 THEN LET Y=25

```

SPECTRUM:

```

1190 DIM B$(8,1)
1220 PRINT AT 5,1;
1240 PRINT AT 5,1;
1260 LET A$=INKEY$:IF A$="" THEN GOTO 1260
1910 IF X>31 THEN LET X=31
1920 IF X<0 THEN LET X=0
1930 IF Y>21 THEN LET Y=21
1940 IF Y<0 THEN LET Y=0

```

Como puede verse en este programa, es el usuario el que indica qué teclas serán las que permitirán el movimiento del asterisco por la pantalla. Incluir una rutina como ésta en todos nuestros programas de juegos (o de otro tipo) nunca viene mal, pues permite que el usuario elija lo que él cree que es mejor para él.

Habréis apreciado que el programa es muy parecido al 3 del tomo anterior. De hecho, es el mismo, pero con las modificaciones necesarias para que se puedan redefinir las teclas. Ahora vamos a ver el funcionamiento del programa. Las partes del programa que no han cambiado y que, por tanto, ya hemos explicado, no las tocaremos.

Línea 1100. Borrarnos la pantalla.

Líneas 1110 a 1180. Imprimimos un mensaje que le dice al usuario la forma en que tiene que utilizar la redefinición de teclas.

Línea 1190. Definimos un vector alfanumérico de ocho elementos. Dentro de cada elemento de dicho vector almacenaremos una de las teclas de movimiento.

Línea 1200. Comienza un bucle, dentro del cual iremos imprimiendo la dirección del movimiento, cogeremos el carácter del teclado que le interesa al usuario y lo almacenaremos.

Línea 1210. Se lee la dirección de las líneas DATA, que se encuentran a partir de la línea 1550.

Línea 1220. Colocamos el cursor en una cierta posición de la pantalla (primera columna de la línea 15).

Línea 1230. Imprimimos 25 caracteres en blanco. Esta impresión no sirve para nada en la primera vuelta del bucle, pero, a partir de la segunda, nos permite borrar la antigua dirección que esté escrita en la pantalla.

Línea 1240. Volvemos a colocar el cursor en la misma posición donde lo colocamos anteriormente.

Línea 1250. E imprimimos la dirección de la cual queremos saber qué tecla ha elegido el usuario para su movimiento.

Línea 1260. Esperamos hasta que el usuario haya pulsado alguna tecla.

Línea 1270. Almacenamos en el vector B\$() la tecla pulsada.

Línea 1280. Imprimimos el carácter igual seguido de la tecla pulsada.

Líneas 1290 y 1300. Se realiza un bucle en vacío para hacer un retardo de tiempo. Esto sirve para que el usuario vea en la pantalla la tecla que ha pulsado y que ha elegido para controlar el movimiento del asterisco en esa dirección.

Línea 1310. Aquí termina el bucle principal.

Línea 1350. Una vez que el usuario ha definido todas las teclas borramos la pantalla.

Líneas 1360 a 1400. Se imprimen una serie de líneas que le dicen al usuario que ahora vamos a ver juntas todas las teclas que él ha definido.

Línea 1410. Hacemos que el puntero que indica la línea DATA de la cual tenemos que leer se vuelva a colocar en la línea 1550. Si no hiciésemos esto, el ordenador nos daría un error, ya que no encontraría más líneas DATA en el resto del programa.

Línea 1420. Comenzamos un bucle, gracias al cual vamos a imprimir todas las direcciones que permite el programa y, junto a cada dirección, la tecla que el usuario ha elegido para controlarla.

Línea 1430. Colocamos el cursor en la primera columna de una línea que depende del valor de la variable índice del bucle (I). En la primera vuelta se colocará en la línea 8. En la segunda en la 9. En la tercera, en la 10, y así sucesivamente.

Línea 1440. Leemos de las líneas DATA la dirección.

Línea 1450. Imprimimos la tecla que controla dicha dirección, un signo igual y la dirección.

Línea 1460. Aquí termina el bucle.

Línea 1470. Dejamos dos líneas en blanco.

Línea 1480. Le preguntamos al usuario si está de acuerdo con las teclas que él mismo ha elegido.

Línea 1490. Y esperamos hasta que éste pulse una tecla.

Línea 1500. Si la tecla pulsada es una N (o una n), entonces volvemos al principio del programa para que el usuario tenga la oportunidad de volver a definir las teclas.

Línea 1510. Si el usuario pulsó alguna tecla distinta de N (y de n), volvemos a la línea 1490 para seguir preguntando hasta que pulse o S o N (también se puede pulsar s o n).

Líneas 1550 a 1580. En estas cuatro líneas se encuentran las ocho direcciones sobre la que el usuario tiene que definir las teclas.

A partir de este momento el resto del programa es igual, salvo por unas pocas líneas. Estas son:

Línea 1710. En esta línea, aparte de mirar si el usuario ha pulsado alguna tecla desde el teclado, se pone a cero la variable SW. Esta nos servirá para, más adelante, saber si la tecla pulsada era una de las permitidas.

Línea 1720. Comenzamos un bucle, dentro del cual veremos si la tecla pulsada por el usuario es una de las permitidas.

Línea 1730. Si la tecla pulsada (A\$) es igual a algunas de las permitidas y que el usuario ha definido (B\$), entonces hacemos que la variable auxiliar SW tenga valor 1.

Línea 1740. Aquí termina el bucle de comprobación.

Línea 1750. Si SW es igual a 0, entonces es que el usuario no pulsó ninguna de las teclas permitidas. En tal caso, devolvemos el control del programa a la línea 1710 para que se recoja una nueva tecla del teclado.

El grupo de líneas comprendido entre la 1810 y la 1880, ambas inclusive, se encargan de saber cuál de las teclas permitidas ha pulsado el usuario. En el tomo anterior estas líneas eran casi iguales, pero en vez de comparar la variable A\$ con otra variable B\$, se comparaba A\$ con una letra específica. Al poner, en vez de teclas específicas, una variable po-

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

demos variar el valor de dicha variable y hacer que este grupo de líneas sirva para cualquier grupo de teclas.

El resto del programa es exactamente igual que lo que estuvimos viendo en el tomo anterior.

En los programas que hemos visto hasta ahora, la estrella sólo se movía cuando el usuario pulsaba una tecla. Si el usuario no pulsaba ninguna tecla, el asterisco se quedaba parado. En muchas ocasiones es necesario que el objeto a mover se esté moviendo continuamente por la pantalla, en la última dirección que se le haya indicado, hasta que el usuario cambie dicha dirección. El programa que vamos a ver a continuación, aparte de dejarnos definir las teclas de movimiento, hace que el asterisco se mueva por la pantalla hasta que nosotros le cambiemos la dirección y el sentido del movimiento pulsando alguna tecla.

```

1000 REM *****
1010 REM * MOVIMIENTO CONTROLADO DE UN *
1020 REM * ASTERISCO. *
1030 REM * *
1040 REM * LAS TECLAS DE MOVIMIENTO *
1050 REM * SON REDEFINIBLES. *
1060 REM *****
1070 REM
1080 REM *** DEFINICION DE LAS TECLAS ***
1090 REM
1100 CLS
1110 PRINT "DEFINICION DE LAS TECLAS"
1120 PRINT "-----"
1130 PRINT
1140 PRINT " Cada vez que aparezca una"
1150 PRINT "direccion en la pantalla ,"
1160 PRINT "pulsas la tecla que quieres"
1170 PRINT "que controle dicha direc. "
1180 PRINT:PRINT:PRINT
1190 DIM B$(8)
1200 FOR I=1 TO 8
1210 READ A$
1220 LOCATE 15,1
1230 PRINT " ":REM 25 ESPACIOS
1240 LOCATE 15,1
1250 PRINT A$;
1260 A$=INKEY$:IF A$="" THEN GOTO 1260
1270 LET B$(I)=A$
1280 PRINT " = ";A$
1290 FOR J=1 TO 200
1300 NEXT J
1310 NEXT I
1320 REM

```

```

1330 REM *** VISUALIZACION DEL LAS TECLAS ***
1340 REM
1350 CLS
1360 PRINT "TECLAS DEFINIDAS"
1370 PRINT "=====
1380 PRINT:PRINT
1390 PRINT "Las teclas definidas son:"
1400 PRINT
1410 RESTORE 1550
1420 FOR I=1 TO 8
1430 LOCATE 7+I,1
1440 READ A$
1450 PRINT B$(I); " = ";A$
1460 NEXT I
1470 PRINT:PRINT
1480 PRINT "ESTAS DE ACUERDO (S/N)"
1490 LET A$=INKEY$:IF A$="" THEN GOTO 1490
1500 IF A$="N" OR A$="n" THEN RUN
1510 IF A$<>"S" AND A$<>"s" THEN GOTO 1490
1520 REM
1530 REM *** LINEAS DE DATA ***
1540 REM
1550 DATA "IZQUIERDA", "DERECHA"
1560 DATA "ARRIBA", "ABAJO"
1570 DATA "ARRIBA-IZQ", "ARRIBA-DER"
1580 DATA "ABAJO-IZQ", "ABAJO-DER"
1590 REM
1600 REM *** PROGRAMA GENERAL ***
1610 REM
1620 CLS
1630 LET X1=10
1640 LET Y1=10
1650 LET X=10
1660 LET Y=10
1670 LOCATE Y1,X1
1680 PRINT " "
1690 LOCATE Y,X
1700 PRINT "*";
1710 LET SW=0:LET A$=INKEY$
1720 FOR I=1 TO 8
1730 IF A$=B$(I) THEN LET SW=1
1740 NEXT I
1750 IF SW=0 THEN LET A$=C$
1760 LET X1=X
1770 LET Y1=Y
1775 LET C$=A$
1780 REM
1790 REM *** ELECCION DEL MOVIMIENTO ***
1800 REM
1810 IF A$=B$(2) THEN LET X=X+1
1820 IF A$=B$(1) THEN LET X=X-1
1830 IF A$=B$(3) THEN LET Y=Y-1
1840 IF A$=B$(4) THEN LET Y=Y+1
1850 IF A$=B$(5) THEN LET X=X-1:LET Y=Y-1
1860 IF A$=B$(6) THEN LET X=X+1:LET Y=Y-1
1870 IF A$=B$(8) THEN LET X=X+1:LET Y=Y+1
1880 IF A$=B$(7) THEN LET X=X-1:LET Y=Y+1
1890 REM
1900 REM
1910 IF X>40 THEN LET X=40
1920 IF X<1 THEN LET X=1
1930 IF Y>24 THEN LET Y=24
1940 IF Y<1 THEN LET Y=1
1950 GOTO 1670

```

Las modificaciones para ordenadores distintos del IBM y compatibles son:

COMMODORE:

```
1100 PRINT CHR$(147)
1220 POKE 214,15:POKE 211,0
1240 POKE 214,15:POKE 211,0
1260 GET A$:IF A$="" THEN GOTO 1260
1350 PRINT CHR$(147)
1430 POKE 214,7+I:POKE 211,0
1490 GET A$:IF A$="" THEN GOTO 1490
1620 PRINT CHR$(147)
1670 POKE 214,Y1:POKE 211,X1
1690 POKE 214,Y:POKE 211,X
1910 IF X>39 THEN LET X=39
1920 IF X<0 THEN LET X=0
1930 IF Y>23 THEN LET Y=23
1940 IF Y<0 THEN LET Y=0
```

MSX:

```
1220 LOCATE 1,15
1240 LOCATE 1,15
1430 LOCATE 1,7+I
1670 LOCATE X1,Y1
1690 LOCATE X,Y
```

AMSTRAD:

```
1220 LOCATE 1,15
1240 LOCATE 1,15
1430 LOCATE 1,7+I
1670 LOCATE X1,Y1
1690 LOCATE X,Y
1930 IF Y>25 THEN LET Y=25
```

SPECTRUM:

```
1105 LET C$=""
1190 DIM B$(8,1)
1220 PRINT AT 5,1;
1240 PRINT AT 5,1;
1260 LET A$=INKEY$:IF A$="" THEN
GOTO 1260
1910 IF X>31 THEN LET X=31
```

```
1920 IF X<0 THEN LET X=0
1930 IF Y>21 THEN LET Y=21
1940 IF Y<0 THEN LET Y=0
```

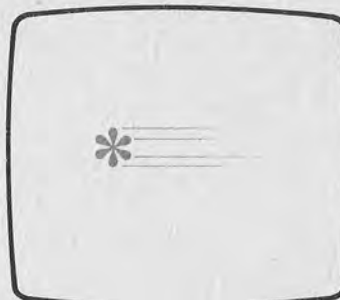


Fig. 2. Con el programa 2, el asterisco se está moviendo continuamente.

Como puede apreciarse, las modificaciones son muy pequeñas. Sólo se ha tenido que variar la línea 1750 e incluir una nueva (la 1775) para que el asterisco, una vez pulsada la primera tecla, esté continuamente moviéndose. Explicaremos ahora el efecto de estos dos cambios.

Cuando el programa se ejecuta, y una vez definidas las teclas, el valor de A\$ y de C\$ es nulo (A\$="" y C\$=""). En cuanto el usuario pulsa una de las teclas permitidas, A\$ pasa a ser dicha tecla. Como ejemplo digamos que pulsó la tecla P (movimiento hacia la derecha). Cuando el programa llega a la línea 1775 se almacena en C\$ el valor de A\$ (en este caso una P). Una vez que el programa ha movido el asterisco una posición más a la derecha y vuelve a la línea 1710. Si no se pulsa ninguna otra tecla, como SW es igual a cero, entonces hacemos que A\$ sea igual a C\$. El valor de C\$ no ha sido cambiado, por lo que sigue siendo una P. A\$ es una P y el programa sigue hacia abajo, con lo que el asterisco se mueve otra posición a la derecha. Este movimiento continúa hasta que el usuario pulsa otra de las teclas permitidas.

Veamos una especie de juego basado en este mismo principio.

```
1000 REM *****
1010 REM *
1020 REM * JUEGO DEL ASTERISCO *
1030 REM *
1040 REM * LAS TECLAS DE MOVIMIENTO *
1050 REM * SON REDEFINIBLES. *
1060 REM *****
1070 REM
1080 REM *** DEFINICION DE LAS TECLAS ***
1090 REM
1100 LET N=0
```

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

```
1110 CLS
1120 PRINT "DEFINICION DE LAS TECLAS"
1130 PRINT "-----"
1140 PRINT
1150 PRINT " Cada vez que aparezca una"
1160 PRINT "direccion en la pantalla ,"
1170 PRINT "pulsas la tecla que quieres"
1180 PRINT "que controle dicha direc. "
1190 PRINT:PRINT:PRINT
1200 DIM B$(4)
1210 FOR I=1 TO 4
1220   READ A$
1230   LOCATE 15,1
1240   PRINT "                                ":REM 25 ESPACIOS
1250   LOCATE 15,1
1260   PRINT A$;
1270   A$=INKEY$:IF A$="" THEN GOTO 1270
1280   LET B$(I)=A$
1290   PRINT " = ";A$
1300   FOR J=1 TO 200
1310     NEXT J
1320 NEXT I
1330 REM
1340 REM *** VISUALIZACION DE LAS TECLAS ***
1350 REM
1360 CLS
1370 PRINT "TECLAS DEFINIDAS"
1380 PRINT "-----"
1390 PRINT:PRINT
1400 PRINT "Las teclas definidas son:"
1410 PRINT
1420 RESTORE 1560
1430 FOR I=1 TO 4
1440   LOCATE 7+I,1
1450   READ A$
1460   PRINT B$(I);" = ";A$
1470 NEXT I
1480 PRINT:PRINT
1490 PRINT "ESTAS DE ACUERDO (S/N)"
1500 LET A$=INKEY$:IF A$="" THEN GOTO 1500
1510 IF A$="N" OR A$="n" THEN RUN
1520 IF A$<>"S" AND A$<>"s" THEN GOTO 1500
1530 REM
1540 REM *** LINEAS DE DATA ***
1550 REM
1560 DATA "IZQUIERDA", "DERECHA"
1570 DATA "ARRIBA", "ABAJO"
1580 REM
1590 REM *** PROGRAMA GENERAL ***
1600 REM
1610 CLS
1620 LET Y=10
1630 LET X=10
1640 DIM A(41,25)
1650 LET A$=B$(2)
1660 GOTO 1710
1670 LOCATE Y,X
1680 PRINT "*";
1690 LET SW=0
1700 LET A$=INKEY$
1710 FOR I=1 TO 4
1720   IF A$=B$(I) THEN LET SW=1
1730 NEXT I
1740 IF SW=0 THEN LET A$=C$
```

```

1750 LET C$=A$
1760 REM
1770 REM *** ELECCION DEL MOVIMIENTO ***
1780 REM
1790 IF A$=B$(2) THEN LET X=X+1
1800 IF A$=B$(1) THEN LET X=X-1
1810 IF A$=B$(3) THEN LET Y=Y-1
1820 IF A$=B$(4) THEN LET Y=Y+1
1830 REM
1840 REM *** COMPROBACION DE CHOQUE ***
1850 REM
1860 IF A(X,Y)=0 THEN GOTO 1900
1870 LOCATE 21,1
1880 PRINT "LO SIENTO HAS CHOCADO"
1890 END
1900 LET A(X,Y)=1
1910 LET N=N+1
1920 REM
1930 REM *** COMPROBACION DE LOS LIMITES DE LA PANTALLA ***
1940 REM
1950 IF X>40 THEN LET X=40
1960 IF X<1 THEN LET X=1
1970 IF Y<1 THEN LET Y=1
1980 IF Y>24 THEN LET Y=24
1990 GOTO 1670

```

Como siempre, las variaciones que hay que hacer en el programa para que funcione en ordenadores distintos del IBM y compatibles son las siguientes:

COMMODORE:

```

1100 PRINT CHR$(147)
1230 POKE 214,15:POKE 211,0
1250 POKE 214,15:POKE 211,0
1270 GET A$:IF A$="" THEN GOTO 1270
1360 PRINT CHR$(147)
1440 POKE 214,7+I:POKE 211,0
1500 GET A$:IF A$="" THEN GOTO 1500
1610 PRINT CHR$(147)
1670 POKE 214,Y:POKE 211,X
1870 POKE 214,21:POKE 211,0
1950 IF X>39 THEN LET X=39
1960 IF X<0 THEN LET X=0
1970 IF Y<0 THEN LET Y=0
1980 IF Y>23 THEN LET Y=23

```

MSX:

```

1230 LOCATE 1,15
1250 LOCATE 1,15
1440 LOCATE 1,7+I
1670 LOCATE X,Y
1870 LOCATE 1,21

```

AMSTRAD:

```

1230 LOCATE 1,15
1250 LOCATE 1,15
1440 LOCATE 1,7+I

```

```

1670 LOCATE X,Y
1870 LOCATE 1,21

```

SPECTRUM:

```

1200 DIM B$(4,1)
1230 PRINT AT 15,1;
1250 PRINT AT 15,1;
1270 LET A$=INKEY$:IF A$="" THEN
GOTO 1270
1440 PRINT AT 7+I,1;
1670 PRINT AT Y,X;
1870 LOCATE 21,1;
1950 IF X>31 THEN LET X=31
1960 IF X<0 THEN LET X=0
1970 IF Y<0 THEN LET Y=0
1980 IF Y>21 THEN LET Y=21

```

Aunque el juego no es ninguna maravilla, ilustra perfectamente todo lo que hemos estudiado hasta ahora. El juego consiste en mover el asterisco por la pantalla el mayor tiempo posible sin chocar con el rastro que va dejando. Para ello podemos movernos hacia arriba, hacia abajo, a la derecha y a la izquierda. Podemos definir las teclas como queramos. En el momento que choquemos con nuestro propio rastro el programa terminará.

La explicación del programa no es necesaria, pues todo lo que en él aparece lo hemos visto anteriormente. De ahí el parecido con los demás programas de este tomo.

Lo único que hay que decir del progra-

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS

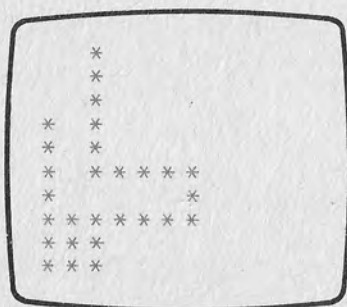


Fig. 3. El programa 3 nos muestra cómo hacer un juego sencillo y divertido.

ma es que, para no tener que mirar en la pantalla si hemos chocado con nosotros mismos, se define una matriz de 41×25 elementos (tantos como caracteres en la pantalla). Como la matriz está dividida en filas y columnas (al igual que la pantalla), cada vez que el asterisco pase por 5 una cierta posición de la pantalla, almacenaremos en la misma posición de la matriz un uno (1). Con esto conseguimos que, tras cada movimiento del asterisco, podamos comprobar si dicha posición de la pantalla ya ha sido ocupada. En caso afirmativo es que nos hemos chocado. En caso negativo nos movemos a la nueva posición y ponemos un uno (1) en la del elemento correspondiente de la matriz.

Una vez entendido cómo funciona este juego, se propone otro que aparecerá en el próximo tomo. Este consistirá en mover un gusano por la pantalla en busca de alimento. El alimento aparecerá de forma aleatoria por la pantalla. Cuanto más alimento coma el gusano, más grande se hará y más difícil será controlarlo sin que se choque consigo mismo. El juego habrá de ser una versión simplificada del famoso SNAKEPIT.

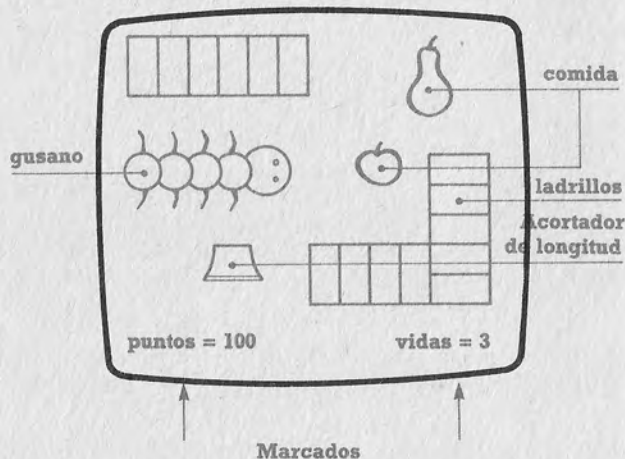
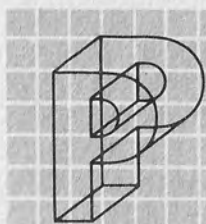


Fig. 4. Estos son algunos de los elementos que puede llevar el programa.

TRUCOS Y RUTINAS BASICAS



OR fin, en este tomo vamos a poder ver el programa gestor de ficheros terminado en su totalidad, pero, debido a su gran extensión, no vamos a poder comentarlo hasta el siguiente tomo. Las modificaciones

necesarias para que el programa funcione en el MSX, AMSTRAD, SPECTRUM y COM-

MODERE también aparecerán en el próximo tomo.

Se ha reenumerado el programa para que quede más compacto y más armonioso en su totalidad. Algunas de las rutinas se han cambiado a última hora para ganar velocidad o bien por haber encontrado algún algoritmo más elegante y bonito, pero la filosofía y la forma de funcionar cada rutina permanece sin alterar.

```
100 REM *****
110 REM * GESTOR DE FICHEROS *
120 REM *****
130 REM
140 REM *****
150 REM * Por Fco. Morales Guerrero *
160 REM *****
170 REM * (c) Ed. Siglo Cultural, 1987 *
180 REM *****
190 REM
200 REM *** INICIALIZACION ***
210 REM
220 DIM F$(1000),A$(10),N(20),N$(20,4)
230 LET TT=0
240 CLS
250 LET A$="MENU INICIAL"
260 LET N=2
270 LET A$(1)="DEFINIR FICHERO NUEVO"
280 LET A$(2)="ENTRAR EN EL MENU GENERAL"
290 LET M$="2"
300 LET W$="1"
310 GOSUB 5960
320 IF A$="1" THEN GOSUB 7250:GOTO 240
330 CLS
340 LET A$="MENU GENERAL"
350 LET N=8
360 LET A$(1)="INTRODUCIR FICHAS"
370 LET A$(2)="GESTIONAR FICHAS"
380 LET A$(3)="ORDENAR FICHAS"
390 LET A$(4)="BUSCAR FICHAS"
400 LET A$(5)="GRABACION"
410 LET A$(6)="LECTURA"
420 LET A$(7)="IMPRESORA"
430 LET A$(8)="TERMINAR"
440 LET M$="8"
```

TRUCOS Y RUTINAS BASICAS

```
450 GOSUB 5960
460 LET A=VAL(A$)
470 ON A GOSUB 490,650,1050,1150,1450,1990,2550,3200
480 GOTO 330
490 REM
500 REM *** INTRODUCIR FICHAS ***
510 REM
520 CLS
530 PRINT " INTRODUCIR FICHAS"
540 PRINT "=====
550 PRINT "Fichas en memoria = ";TT
560 LET NF=TT+1:LET F$(NF)=CHR$(254)
570 GOSUB 8010
580 LET TT=TT+1
590 LOCATE 20,1
600 PRINT "INTRODUCIR OTRA FICHA (S/N)"
610 LET A$=INKEY$:IF A$="" THEN GOTO 610
620 IF A$="S" OR A$="s" THEN GOTO 490
630 IF A$="N" OR A$="n" THEN RETURN
640 GOTO 610
650 REM
660 REM *** GESTIONAR FICHAS ***
670 REM
680 LET A$="GESTION DE FICHAS"
690 LET N=4
700 LET A$(1)="BORRAR FICHA"
710 LET A$(2)="LIMPIEZA DEL FICHERO"
720 LET A$(3)="MODIFICAR FICHA"
730 LET A$(4)="VER UNA FICHA"
740 GOSUB 5960
750 CLS
760 ON VAL(A$) GOTO 770,850,890,980
770 PRINT "BORRAR FICHA"
780 PRINT "=====
790 PRINT
800 IF N(1)=0 THEN PRINT "NO HAY FICHAS":GOSUB 5440:RETURN
810 INPUT "QUE No. DE FICHA ";NF
820 IF NF<1 OR NF>77 THEN GOTO 810
830 LET F$(NF)=CHR$(253)+F$(NF)
840 RETURN
850 PRINT "LIMPIAR FICHERO"
860 PRINT "=====
870 GOSUB 5680
880 RETURN
890 PRINT "MODIFICAR FICHA"
900 PRINT "=====
910 PRINT
920 INPUT "QUE FICHA MODIFICAS ";NF
930 IF NF<1 OR NF>TT THEN PRINT "ESA FICHA NO EXISTE":GOSUB 5440:RETURN
940 GOSUB 8290
950 CLS
960 GOSUB 8010
970 RETURN
980 PRINT "VER UNA FICHA"
990 PRINT "=====
1000 PRINT
1010 INPUT "QUE FICHA QUIERES VER ";NF
1020 IF NF<1 OR NF>TT THEN PRINT "ESA FICHA NO EXISTE":GOSUB 5440:RETURN
1030 GOSUB 8290
1040 RETURN
1050 REM
1060 REM *** ORDENAR FICHAS ***
1070 REM
1080 CLS
```

```

1090 PRINT " ORDENAR FICHAS"
1100 PRINT "-----"
1110 PRINT:PRINT:PRINT
1120 INPUT "ORDENAMOS POR EL CAMPO No. :";NC
1130 GOSUB 6890
1140 RETURN
1150 REM
1160 REM *** BUSCAR FICHAS ***
1170 REM
1180 PRINT "BUSQUEDA DE FICHAS"
1190 PRINT "=====
1200 PRINT:PRINT
1210 IF N(1)=0 THEN PRINT "NO HAY FICHAS":GOSUB 5440:RETURN
1220 LET A$="BUSQUEDA DE FICHAS"
1230 LET N=3
1240 LET A$(1)="VER FICHAS"
1250 LET A$(2)="BUSCAR POR CAMPO Y VALOR"
1260 LET A$(3)="BUSCAR FICHAS BORRADAS"
1270 LET W$="1"
1280 LET M$="3"
1290 GOSUB 5960
1300 CLS
1310 PRINT "BUSQUEDA DE FICHAS"
1320 PRINT "=====
1330 PRINT:PRINT:LET SW=VAL(A$)-1
1340 ON VAL(A$) GOTO 1350,1390,1430
1350 INPUT "INTRODUCE PRIMERA FICHA ";N1
1360 PRINT
1370 INPUT "INTRODUCE ULTIMA FICHA ";N2
1380 GOTO 1430
1390 INPUT "QUE VALOR HAY QUE BUSCAR ";A$
1400 PRINT
1410 INPUT "EN QUE CAMPO ";NC
1420 GOTO 1430
1430 GOSUB 6230
1440 RETURN
1450 REM
1460 REM *** GRABACION ***
1470 REM
1480 LET A$="GRABACION DEL FICHERO"
1490 LET N=2
1500 LET A$(1)="GRABACION EN DISCO"
1510 LET A$(2)="GRABACION EN CINTA"
1520 LET W$="1"
1530 LET M$="2"
1540 GOSUB 5960
1550 CLS
1560 PRINT "GRABACION DEL FICHERO"
1570 PRINT "=====
1580 PRINT:PRINT
1590 INPUT "NOMBRE DEL FICHERO ";B$
1600 IF LEN(B$)>12 THEN GOTO 1590
1610 PRINT:PRINT
1620 PRINT "GRABANDO ";B$
1630 IF A$="2" THEN GOTO 1810
1640 OPEN B$ FOR OUTPUT AS #1
1650 PRINT #1,N(1)
1660 FOR Z=1 TO N(1)
1670   PRINT #1,N(Z+1)
1680 NEXT Z:FOR Z=1 TO N(1)
1690   PRINT #1,M$(Z)
1700 NEXT Z:PRINT #1,TT
1710 FOR Z=1 TO TT
1720   PRINT #1,F$(Z)

```

TRUCOS Y RUTINAS BASICAS

```
1730 NEXT Z
1740 FOR Z=1 TO 20
1750   FOR X=1 TO 4
1760     PRINT #1,N$(Z,X)
1770   NEXT X
1780 NEXT Z
1790 CLOSE #1
1800 RETURN
1810 OPEN "CAS:"+B$ FOR OUTPUT AS #1
1820 PRINT #1,N(1)
1830 FOR Z=1 TO N(1)
1840   PRINT #1,N(Z+1)
1850 NEXT Z
1860 FOR Z=1 TO N(1)
1870   PRINT #1,M$(Z)
1880 NEXT Z
1890 FOR Z=1 TO TT
1900   PRINT #1,F$(Z)
1910 NEXT Z
1920 FOR Z=1 TO 20
1930   FOR X=1 TO 4
1940     PRINT #1,N$(Z,X)
1950   NEXT X
1960 NEXT Z
1970 CLOSE #1
1980 RETURN
1990 REM
2000 REM *** LECTURA DEL FICHERO ***
2010 REM
2020 LET A$="LECTURA DE UN FICHERO"
2030 LET N=2
2040 LET A$(1)="LECTURA DESDE DISCO"
2050 LET A$(2)="LECTURA DESDE CASETE"
2060 LET W$="1"
2070 LET M$="2"
2080 GOSUB 5960
2090 CLS
2100 PRINT "GRABACION DEL FICHERO"
2110 PRINT "===== "
2120 PRINT:PRINT
2130 INPUT "NOMBRE DEL FICHERO ";B$
2140 IF LEN(B$)>12 THEN GOTO 2130
2150 PRINT "LEYENDO ... ";B$
2160 IF A$="2" THEN GOTO 2360
2170 OPEN B$ FOR INPUT AS #1
2180 INPUT #1,N(1)
2190 FOR Z=1 TO N(1)
2200   INPUT #1,N(Z+1)
2210 NEXT Z
2220 FOR Z=1 TO N(1)
2230   INPUT #1,M$(Z)
2240 NEXT Z
2250 INPUT #1,TT
2260 FOR Z=1 TO TT
2270   INPUT #1,F$(Z)
2280 NEXT Z
2290 FOR Z=1 TO 20
2300   FOR X=1 TO 4
2310     INPUT #1,N$(Z,X)
2320   NEXT X
2330 NEXT Z
2340 CLOSE #1
2350 RETURN
2360 OPEN "CAS"+B$ FOR INPUT AS #1
```

```

2370 INPUT #1,N(1)
2380 FOR Z=1 TO N(1)
2390     INPUT #1,N(Z+1)
2400 NEXT Z
2410 FOR Z=1 TO N(1)
2420     INPUT #1,M$(Z)
2430 NEXT Z
2440 INPUT #1,TT
2450 FOR Z=1 TO TT
2460     INPUT #1,F$(Z)
2470 NEXT Z
2480 FOR Z=1 TO 20
2490     FOR X=1 TO 4
2500         INPUT #1,N$(Z,X)
2510     NEXT X
2520 NEXT Z
2530 CLOSE #1
2540 RETURN
2550 REM
2560 REM *** IMPRESORA ***
2570 REM
2580 LET A$="MENU DE IMPRESION"
2590 LET N=2
2600 LET A$(1)="DEFINIR IMPRESION"
2610 LET A$(2)="IMPRIMIR"
2620 LET W$="1"
2630 LET M$="2"
2640 GOSUB 5960
2650 IF A$="1" THEN GOSUB 3390:RETURN
2660 CLS
2670 PRINT "IMPRESION DE LAS FICHAS"
2680 PRINT "=====
2690 PRINT:PRINT
2700 IF N$(1,1)="" THEN PRINT "IMPRESION NO DEFINIDA":GOSUB 5440:RETURN
2710 IF N(1)=0 THEN PRINT "NO HAY FICHAS":GOSUB 5440:RETURN
2720 PRINT "PREPARA LA IMPRESORA Y ..."
2730 GOSUB 5440
2740 LOCATE 5,1
2750 PRINT "IMPRIMIREMOS LAS FICHAS ";
2760 FOR Z=1 TO 20
2770     IF N$(Z,1)<>"" THEN PRINT N$(Z,1);", ";:GOTO 2790
2780     LET Z=21:LET N=Z
2790 NEXT Z
2800 PRINT:PRINT
2810 PRINT "EN EL ORDEN ";
2820 FOR Z=1 TO N
2830     IF N$(Z,2)<>"" THEN PRINT N$(Z,2);", ";:GOTO 2850
2840     LET Z=21
2850 NEXT Z
2860 PRINT:PRINT
2870 PRINT "TABULADAS EN LAS POSICIONES ";
2880 FOR Z=1 TO N
2890     IF N$(Z,3)<>"" THEN PRINT N$(Z,3);", ";:GOTO 2910
2900     LET Z=21
2910 NEXT Z
2920 PRINT:PRINT
2930 PRINT "ENTRE FICHA Y FICHA ";
2940 IF N$(1,4)="-1" THEN PRINT "CAMBIARE DE PAGINA":GOTO 2960
2950 PRINT "IMPRIMIRE ";N$(1,4);" LINEAS EN BLANCO"
2960 GOSUB 3150
2970 LET CC=0
2980 FOR Z=1 TO TT:LET CR=1
2990     FOR R=1 TO 20:LET CD=0
3000         IF N$(R,2)="" THEN LET R=20:GOTO 3090

```

TRUCOS Y RUTINAS BASICAS

```

3010     LET CC=VAL(N$(R,2))
3020     FOR V=1 TO LEN(F$(Z))
3030         IF MID$(F$(Z),V,1)=CHR$(254) THEN LET CD=CD+1:IF CD=CC THEN LET Z1
=V+1:LET V=LEN(F$(Z))
3040     NEXT V:LPRINT TAB(VAL(N$(CR,3))):LET CR=CR+1
3050     FOR V=Z1 TO LEN(F$(Z))
3060         IF MID$(F$(Z),V,1)=CHR$(254) THEN LET V=LEN(F$(Z)):GOTO 3080
3070         LPRINT MID$(F$(Z),V,1);
3080     NEXT V
3090 NEXT R
3100 IF N$(1,4)="-1" THEN LPRINT CHR$(12):GOSUB 3150:GOTO 3120
3110 FOR R=1 TO VAL(N$(1,4)):LPRINT:NEXT R
3120 NEXT Z
3130 LPRINT "-----"
-----"
3140 RETURN
3150 FOR S=1 TO 20
3160     IF N$(S,2)="" THEN GOTO 3180
3170     LPRINT TAB(VAL(N$(S,3))):M$(VAL(N$(S,2)));
3180 NEXT S:LPRINT "-----"
-----"
3190 RETURN
3200 REM
3210 REM *** FIN DE LA SESION ***
3220 REM
3230 CLS
3240 PRINT "TERMINAR LA SESION"
3250 PRINT "=====
3260 PRINT:PRINT:PRINT
3270 PRINT "ESTA SEGURO (S/N)"
3280 LET A$=INKEY$
3290 IF A$="" THEN GOTO 3280
3300 IF A$="S" OR A$="s" THEN GOTO 3330
3310 IF A$="N" OR A$="n" THEN RETURN
3320 GOTO 3280
3330 CLS
3340 PRINT "SESION TERMINADA. ADIOS"
3350 FOR I=1 TO 10
3360     PRINT
3370 NEXT I
3380 END
3390 REM
3400 REM *****
3410 REM * DEFINICION DE IMPRESION *
3420 REM *****
3430 REM
3440 CLS
3450 PRINT "DEFINICION DE LA IMPRESION"
3460 PRINT "=====
3470 LOCATE 5,1
3480 PRINT "( CUANTOS CAMPOS QUIERE QUE IMPRIMA ?"
3490 LOCATE 12,1
3500 LINE INPUT "=> ";A$
3510 IF VAL(A$)=0 THEN RETURN
3520 LET N=VAL(A$)
3530 LOCATE 5,1
3540 PRINT "( QUE CAMPOS DESEA QUE IMPRIMA ?
3550 LOCATE 9,1
3560 PRINT "CAMPOS A IMPRIMIR = "
3570 FOR Z=1 TO N
3580     LOCATE 12,1
3590     PRINT SPACE$(10)
3600     LOCATE 12,1

```

```

3610 LINE INPUT "=> ";N$(Z,1)
3620 IF VAL(N$(Z,1))=0 THEN GOTO 3580
3630 LOCATE 9,22
3640 FOR X=1 TO Z
3650 PRINT N$(X,1);",";
3660 NEXT X
3670 NEXT Z
3680 PRINT CHR$(29);" "
3690 LOCATE 5,1
3700 PRINT "( EN QUE ORDEN LOS IMPRIMO ? "
3710 PRINT "INTRODUZCA EL NUMERO DEL CAMPO EN EL "
3720 PRINT "ORDEN EN QUE QUIERE QUE SE IMPRIMA. "
3730 LOCATE 10,1
3740 PRINT "ORDEN DE IMPRESION = "
3750 FOR Z=1 TO N
3760 LOCATE 12,1
3770 PRINT SPACE$(10)
3780 LOCATE 12,1
3790 LINE INPUT "=> ";N$(Z,2)
3800 IF VAL(N$(Z,2))=0 THEN GOTO 3760
3810 LOCATE 10,22
3820 FOR X=1 TO Z
3830 PRINT N$(X,2);",";
3840 NEXT X
3850 NEXT Z
3860 PRINT CHR$(29);" "
3870 LOCATE 12,1
3880 PRINT " " " :REM 10 ESPACIOS EN BLANCO
3890 LET NN=0
3900 FOR Z=1 TO N
3910 FOR X=1 TO N
3920 IF N$(Z,1)=N$(X,2) THEN LET NN=NN+1:LET X=N+1
3930 NEXT X
3940 NEXT Z
3950 IF NN=N THEN GOTO 4150
3960 CLS
3970 PRINT "CAMPOS A IMPRIMIR = ";
3980 FOR Z=1 TO N
3990 PRINT N$(Z,1);",";
4000 NEXT Z
4010 PRINT CHR$(29);" "
4020 PRINT
4030 PRINT
4040 PRINT "ORDEN DE IMPRESION = ";
4050 FOR Z=1 TO N
4060 PRINT N$(Z,2);",";
4070 NEXT Z
4080 PRINT CHR$(29);" "
4090 LOCATE 12,1
4100 PRINT "NO COINCIDEN LOS CAMPOS A IMPRIMIR"
4110 PRINT "CON EL ORDEN DE IMPRESION"
4120 LET X=1:LET Y=20
4130 GOSUB 5440
4140 GOTO 3390
4150 LOCATE 15,1
4160 PRINT "( ESTA DE ACUERDO ? (S/N)"
4170 LET A$=INKEY$
4180 IF A$="S" OR A$="s" THEN GOTO 4220
4190 IF A$<>"N" AND A$<>"n" THEN GOTO 4170
4200 ERASE N$
4210 GOTO 3390
4220 FOR Z=5 TO 20
4230 LOCATE Z,1
4240 PRINT SPACE$(40)

```

TRUCOS Y RUTINAS BASICAS

```

4250 NEXT Z
4260 LOCATE 5,1
4270 PRINT "INTRODUZCA AHORA EN QUE COLUMNA"
4280 PRINT "DE LA IMPRESORA SE HA DE IMPRI-"
4290 PRINT "MIR CADA CAMPO."
4300 FOR Z=1 TO N
4310     LOCATE 12,1
4320     PRINT "CAMPO No. ";N$(Z,1);" ";
4330     LINE INPUT "=="> ";N$(Z,3)
4340     IF VAL(N$(Z,3))=0 THEN GOTO 4310
4350 NEXT Z
4360 FOR Z=5 TO 17
4370     LOCATE Z,1
4380     PRINT SPACE$(40)
4390 NEXT Z
4400 LOCATE 5,1
4410 FOR Z=1 TO N
4420     PRINT "EL CAMPO ";N$(Z,1);" SE TABULARA EN LA COLUMNA ";N$(Z,3)
4430 NEXT Z
4440 LOCATE 20,1
4450 PRINT "( ESTA DE ACUERDO ? (S/N)"
4460 LET A$=INKEY$
4470 IF A$="S" OR A$="s" THEN GOTO 4500
4480 IF A$<>"N" AND A$<>"n" THEN GOTO 4460
4490 GOTO 4220
4500 FOR Z=5 TO 20
4510     LOCATE Z,1
4520     PRINT SPACE$(40)
4530 NEXT Z
4540 LOCATE 5,1
4550 PRINT "( CUANTAS LINEAS DEJO ENTRE FICHA"
4560 PRINT "Y FICHA? PULSA -1 PARA CAMBIAR DE"
4570 PRINT "PAGINA."
4580 LOCATE 12,1
4590 LINE INPUT "=="> ";N$(1,4)
4600 IF VAL(N$(1,4))=0 THEN GOTO 4580
4610 CLS
4620 PRINT "OPERACION COMPLETADA"
4630 PRINT "====="
4640 LET X=1:LET Y=20
4650 GOSUB 5440
4660 CLS
4670 RETURN
4680 REM
4690 REM *****
4700 REM * SUBROUTINA DE ENTRADA DE DATOS *
4710 REM *****
4720 REM
4730 LET D$="" : LET LO=0
4740 LOCATE Y,X
4750 FOR Z=1 TO LO
4760     PRINT ". ";
4770 NEXT Z
4780 LOCATE Y,X
4790 PRINT " _ "
4800 LOCATE Y,X
4810 LET A$=INKEY$
4820 IF A$="" THEN GOTO 4810
4830 IF A$=CHR$(8) AND LO>0 THEN LET LO=LO-1:LET D$=LEFT$(D$,LO) :LET X=X-1:LOCA
TE Y,X:PRINT " _ ":LOCATE Y,X:GOTO 4810
4840 IF A$=" " OR A$="." OR A$="-" THEN GOTO 4870
4850 IF A$=CHR$(13) THEN GOTO 4930
4860 IF A$>M$ OR A$<W$ THEN GOTO 4810
4870 PRINT A$;" _ "

```

```

4880 LET X=X+1
4890 LOCATE Y,X
4900 LET D$=D$+A$
4910 LET LO=LO+1
4920 IF LO<>LO THEN GOTO 4810
4930 LOCATE Y,X
4940 FOR Z=LO TO LO
4950   PRINT " ";
4960 NEXT Z
4970 RETURN
4980 REM
4990 REM *****
5000 REM * MENSAJE Y RECOGIDA DE OPCION *
5010 REM *****
5020 REM
5030 LET T$="INTRODUZCA OPCION"
5040 LET L=17
5050 LOCATE Y,X
5060 PRINT T$;
5070   LOCATE Y,X+19
5080   PRINT CHR$(177);
5090   LET A$=INKEY$
5100   IF A$<>" " AND (A$<W$ OR A$>M$) THEN GOSUB 5250
5110   IF A$<>" " THEN GOSUB 5360:GOTO 5220
5120   FOR Z=1 TO 100
5130     NEXT Z
5140   LOCATE Y,X+19
5150   PRINT " ";
5160   LET A$=INKEY$
5170   IF A$<>" " AND (A$<W$ OR A$>M$) THEN GOSUB 5250
5180   IF A$<>" " THEN GOSUB 5360:GOTO 5220
5190   FOR Z=1 TO 100
5200     NEXT Z
5210 GOTO 5070
5220 LOCATE Y,X
5230 PRINT SPACE$(L+3);
5240 RETURN
5250 REM
5260 REM *** ERROR ***
5270 REM
5280 LOCATE Y,X+19
5290 PRINT "ERROR ..."
5300 FOR Z=1 TO 500
5310   NEXT Z
5320 LOCATE Y,X+19
5330 PRINT "          ":REM 9 ESPACIOS
5340 LET A$=""
5350 RETURN
5360 REM
5370 REM *** IMPRESION DE LA OPCION ELEGIDA ***
5380 REM
5390 LOCATE Y,X+19
5400 PRINT A$;
5410 FOR Z=1 TO 500
5420   NEXT Z
5430 RETURN
5440 REM
5450 REM *****
5460 REM * PULSA UNA TECLA *
5470 REM *****
5480 REM
5490 LET A$="PULSA UNA TECLA"
5500 LET L=15
5510   BEEP

```

TRUCOS Y RUTINAS BASICAS

```

5520   FOR Z=1 TO 6
5530       LOCATE Y,X
5540       PRINT A$;
5550       IF INKEY$<>" " THEN GOTO 5650
5560       FOR K=1 TO 100
5570           NEXT K
5580       LOCATE Y,X
5590       PRINT SPACE$(L);
5600       IF INKEY$<>" " THEN GOTO 5650
5610       FOR K=1 TO 100
5620           NEXT K
5630   NEXT Z
5640 GOTO 5510
5650 LOCATE Y,X
5660 PRINT SPACE$(L);
5670 RETURN
5680 REM
5690 REM *****
5700 REM * SUBROUTINA DE BORRADO REAL DE FICHAS *
5710 REM *****
5720 REM
5730 LET N=0
5740 FOR Z=1 TO TT
5750     IF LEFT$(F$(Z),1)<>CHR$(253) THEN GOTO 5810
5760     LET F$(Z)=" "
5770     LET N=N+1
5780     FOR X=Z TO TT-1
5790         LET F$(X)=F$(X+1)
5800     NEXT X
5810 NEXT Z
5820 FOR Z=TT-N+1 TO TT
5830     LET F$(Z)=" "
5840 NEXT Z
5850 LET TT=TT-N
5860 RETURN
5870 PRINT "E R R O R"
5880 PRINT "-----"
5890 PRINT
5900 PRINT "Lo siento pero no he encontrado el campo No. ";NC
5910 PRINT
5920 PRINT "Estas seguro de que las fichas de este fichero tienen";NC;"campos?"
5930 X=1:Y=20:GOSUB 5440
5940 CLS
5950 RETURN
5960 REM
5970 REM *****
5980 REM * SUBROUTINA DE CREACION DE MENUS *
5990 REM *****
6000 REM
6010 CLS
6020 LET X=INT((40-LEN(A$))/2)
6030 LOCATE 1,X
6040 PRINT A$
6050 LOCATE 2,X
6060 FOR Z=1 TO LEN(A$)
6070     PRINT "-";
6080 NEXT Z
6090 PRINT:PRINT
6100 LET X=0
6110 FOR Z=1 TO N
6120     IF X<LEN(A$(Z)) THEN LET X=LEN(A$(Z))
6130 NEXT Z
6140 LET X=INT((40-X)/2)
6150 FOR Z=1 TO N

```

```

6160 LOCATE 2*Z+3,X
6170 PRINT Z;"-";A$(Z)
6180 NEXT Z
6190 LET X=1:LET Y=21
6200 GOSUB 4980
6210 CLS
6220 RETURN
6230 REM
6240 REM *****
6250 REM * SUBROUTINA DE BUSQUEDA *
6260 REM *****
6270 REM
6280 CLS
6290 PRINT "B U S C A N D O"
6300 PRINT "===== "
6310 PRINT
6320 IF SW=0 THEN PRINT "Buscando las fichas desde la ";N1;" hasta la ";N2
6330 IF SW=1 THEN PRINT "Buscando las fichas que tengan ";A$;" en el campo ";NC
6340 IF SW=2 THEN PRINT "Buscando fichas borradas"
6350 PRINT
6360 PRINT "ESPERA UN MOMENTO"
6370 FOR Z=1 TO 1000
6380 NEXT Z
6390 CLS
6400 ON SW+1 GOSUB 6490,6580,6770
6410 CLS
6420 PRINT "OPERACION TERMINADA"
6430 PRINT "===== "
6440 PRINT
6450 PRINT "No se han encontrado mas fichas."
6460 LET X=1:LET Y=20
6470 GOSUB 5440
6480 RETURN
6490 REM
6500 REM *** BUSQUEDA POR No. DE FICHA ***
6510 REM
6520 FOR R=N1 TO N2
6530 LET NF=R
6540 GOSUB 8290
6550 CLS
6560 NEXT R
6570 RETURN
6580 REM
6590 REM *** BUSQUEDA POR CAMPO Y VALOR ***
6600 REM
6610 IF N(1)<>0 THEN GOTO 6690
6620 PRINT "E R R O R"
6630 PRINT "-----"
6640 PRINT
6650 PRINT "Lo siento. El campo ";NC;" no existe."
6660 GOSUB 5440
6670 CLS
6680 RETURN
6690 LET J$=A$:FOR Z=1 TO TT:LET CD=0:LET CR=0:LET Z1=0
6700 FOR X=1 TO LEN(F$(Z)):IF MID$(F$(Z),X,1)=CHR$(254) THEN LET CD=CD+1:IF C
D=NC THEN LET Z1=X+1:LET X=LEN(F$(Z))
6710 NEXT X:FOR X=Z1 TO LEN(F$(Z)):IF MID$(J$,X-Z1+1,1)=MID$(F$(Z),X,1) THEN
LET CR=CR+1:IF CR=LEN(J$) THEN LET X=LEN(F$(Z))
6720 NEXT X
6730 IF CR=LEN(J$) THEN LET VV=Z:LET NF=Z:GOSUB 8290:LET Z=VV
6740 NEXT Z
6750 RETURN
6760 CLS
6770 REM

```

TRUCOS Y RUTINAS BASICAS

```

6780 REM *** BUSQUEDA DE FICHAS BORRADAS ***
6790 REM
6800 FOR Z=1 TO TT
6810 IF LEFT$(F$(Z),1)<>CHR$(253) THEN GOTO 6870
6820 LET F$(Z)=RIGHT$(F$(Z),LEN(F$(Z))-1)
6830 LET NF=Z
6840 GOSUB 8290
6850 CLS:LET Z=NF
6860 LET F$(Z)=CHR$(253)+F$(Z)
6870 NEXT Z
6880 RETURN
6890 REM
6900 REM *****
6910 REM * ORDENACION ALFANUMERICA *
6920 REM *****
6930 REM
6940 CLS
6950 PRINT "O R D E N A R F I C H E R O"
6960 PRINT "=====
6970 PRINT
6980 PRINT "Ordenando fichero por el campo No. ";NC
6990 PRINT
7000 PRINT "ESPERA UN MOMENTO"
7010 IF N(1)>=NC THEN GOTO 7120
7020 CLS
7030 PRINT "E R R O R"
7040 PRINT "-----
7050 PRINT
7060 PRINT "Lo siento pero no he encontrado el campo No. ";NC
7070 PRINT
7080 PRINT "Estas seguro de que las fichas de este fichero tienen";NC;"campos?"
7090 X=1:Y=20:GOSUB 5440
7100 CLS
7110 RETURN
7120 FOR Z=1 TO TT:LET CC=0:LET CD=0
7130 FOR X=1 TO Z:FOR R=1 TO LEN(F$(Z)):IF MID$(F$(Z),R,1)=CHR$(254) THEN LET
CC=CC+1:IF CC=NC THEN LET Z1=R+1:LET R=LEN(F$(Z))
7140 NEXT R:LET CC=0:FOR R=1 TO LEN(F$(X)):IF MID$(F$(X),R,1)=CHR$(254) THEN
LET CC=CC+1:IF CC=NC THEN LET Z2=R+1:LET R=LEN(F$(X))
7150 NEXT R:IF MID$(F$(Z),Z1)<MID$(F$(X),Z2) THEN LET A$=F$(Z):LET F$(Z)=F$(X
):LET F$(X)=A$
7160 NEXT X
7170 NEXT Z
7180 CLS:PRINT "O P E R A C I O N T E R M I N A D A"
7190 PRINT "=====
7200 PRINT
7210 PRINT "El fichero esta ordenado por el campo No. ";NC
7220 X=1:Y=20:GOSUB 5440
7230 CLS
7240 RETURN
7250 REM
7260 REM *****
7270 REM * DEFINICION DE UN FICHERO NUEVO *
7280 REM *****
7290 REM
7300 CLS
7310 LET A$="CREACION DE UN FICHERO"
7320 LET N=2
7330 LET A$(1)="UTILIZAR UNO QUE ESTE CREADO"
7340 LET A$(2)="CREAR UNO NUEVO"
7350 GOSUB 5960
7360 IF A$="1" THEN GOTO 7640
7370 PRINT " CREACION DE UN NUEVO FICHERO"
7380 PRINT "-----

```

```

7390 PRINT:PRINT
7400 PRINT " Por favor, responda a las preguntas quele van a aparecer a continua
cion."
7410 PRINT "=====
7420 LOCATE 9,1
7430 INPUT "Cuantos campos tendra cada ficha?";N(1)
7440 IF N(1)<0 OR N(1)>20 THEN GOTO 7420
7450 DIM M$(N(1)+1)
7460 PRINT
7470 FOR Z=1 TO N(1)
7480     LOCATE 16,1
7490     PRINT "
7500     LOCATE 18,1
7510     PRINT "
7520     LOCATE 16,1
7530     PRINT "Cuantos caracteres ocupa el campo ";Z
7540     INPUT N(Z+1)
7550     LOCATE 18,1
7560     PRINT "Como se llama este campo"
7570     INPUT M$(Z)
7580 NEXT Z
7590 CLS
7600 PRINT "OPERACION TERINADA"
7610 LET X=1:LET Y=20
7620 GOSUB 5440
7630 RETURN
7640 REM
7650 REM *** UTILIZACION DE UN FICHERO YA CREADO ***
7660 REM
7670 CLS
7680 PRINT " CREACION DE UN FICHERO".
7690 PRINT "=====
7700 PRINT
7710 PRINT " Teclea el nombre del fichero que ya tiene una estructura definida y
del cual laquieres copiar."
7720 PRINT:PRINT:PRINT
7730 INPUT "NOMBRE = ";N$
7740 PRINT:PRINT
7750 PRINT " Asegurate de que esta colocada la cintao disco que contiene dicho f
ichero"
7760 LET X=1:LET Y=20
7770 GOSUB 5440
7780 CLS
7790 PRINT "LEYENDO ... ";N$
7800 OPEN N$ FOR INPUT AS #1
7810 INPUT#1,T
7820 DIM M$(T+1)
7830 FOR Z=1 TO T
7840     INPUT#1,N(Z+1)
7850 NEXT Z
7860 LET N(1)=T
7870 FOR Z=1 TO T
7880     INPUT#1,M$(Z)
7890 NEXT Z
7900 CLOSE#1:CLS
7910 PRINT "CONFIGURACION LEIDA"
7920 PRINT:PRINT
7930 PRINT "Cada ficha tiene ";T;" campos"
7940 PRINT
7950 FOR Z=1 TO T
7960     PRINT "el campo ";Z;" se llama ";M$(Z)
7970 NEXT Z
7980 LET X=1:LET Y=21
7990 GOSUB 5440

```

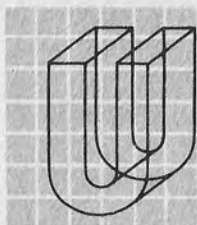
TRUCOS Y RUTINAS BASICAS

```
8000 RETURN
8010 REM
8020 REM *****
8030 REM * INTRODUCCION DE FICHAS *
8040 REM *****
8050 REM
8060 PRINT
8070 PRINT "INTRUDUCIENDO LA FICHA ";NF
8080 PRINT
8090 FOR R=1 TO N(1)
8100   LOCATE 10,1
8110   PRINT M$(R);" : ";
8120   LET X=LEN(M$(R))+4
8130   LET Y=10
8140   LET M$="z"
8150   LET W$="o"
8160   LO=N(R+1):GOSUB 4680
8170   LOCATE 20,1
8180   PRINT "ESTA CORRECTO ESTE CAMPO (S/N)"
8190   LET A$=INKEY$:IF A$="" THEN GOTO 8190
8200   IF A$="N" OR A$="n" THEN GOTO 8100
8210   IF A$<>"S" AND A$<>"s" THEN GOTO 8190
8220   LET F$(NF)=F$(NF)+D$+CHR$(254)
8230   LOCATE 10,1
8240   PRINT SPACE$(80)
8250   LOCATE 20,1
8260   PRINT SPACE$(40)
8270 NEXT R
8280 RETURN
8290 REM
8300 REM *****
8310 REM * VER UNA FICHA *
8320 REM *****
8330 REM
8340 CLS:LET CC=1
8350 PRINT "FICHA No. ";NF
8360 PRINT:PRINT M$(CC);" ";
8370 IF LEFT$(F$(NF),1)=CHR$(253) THEN PRINT "ESTA FICHA ESTA BORRADA":GOSUB 544
8380 FOR Z=2 TO LEN(F$(NF))
8390   IF MID$(F$(NF),Z,1)=CHR$(254) THEN LET CC=CC+1:PRINT:PRINT M$(CC);" ";:G
8400   PRINT MID$(F$(NF),Z,1);
8410 NEXT Z
8420 GOSUB 5440
8430 RETURN
```

EL TALLER DEL HARDWARE

REALIZACION PRACTICA DEL GRABADOS DE EPROMS

■ Circuito completo de grabador



adjunto describe una aplicación para el IBM PC.

TILIZANDO los bloques descritos para realizar las funciones de lectura y grabación de EPROMS con la interfaz de puertos del tipo empleado en la tarjeta de ampliación de puertos de entrada/salida el esquema

Al poseer suficiente corriente de 5 voltios se ha suprimido la generación de este nivel de tensión. La tensión de grabación se obtiene a partir de los 12 voltios mediante un oscilador, basado en el LM555 y un multiplicador de tensión. El control de la tensión de grabación se obtiene mediante un regulador de tipo LM317 y las resistencias de programación controladas por bits del puerto de control y transistores.

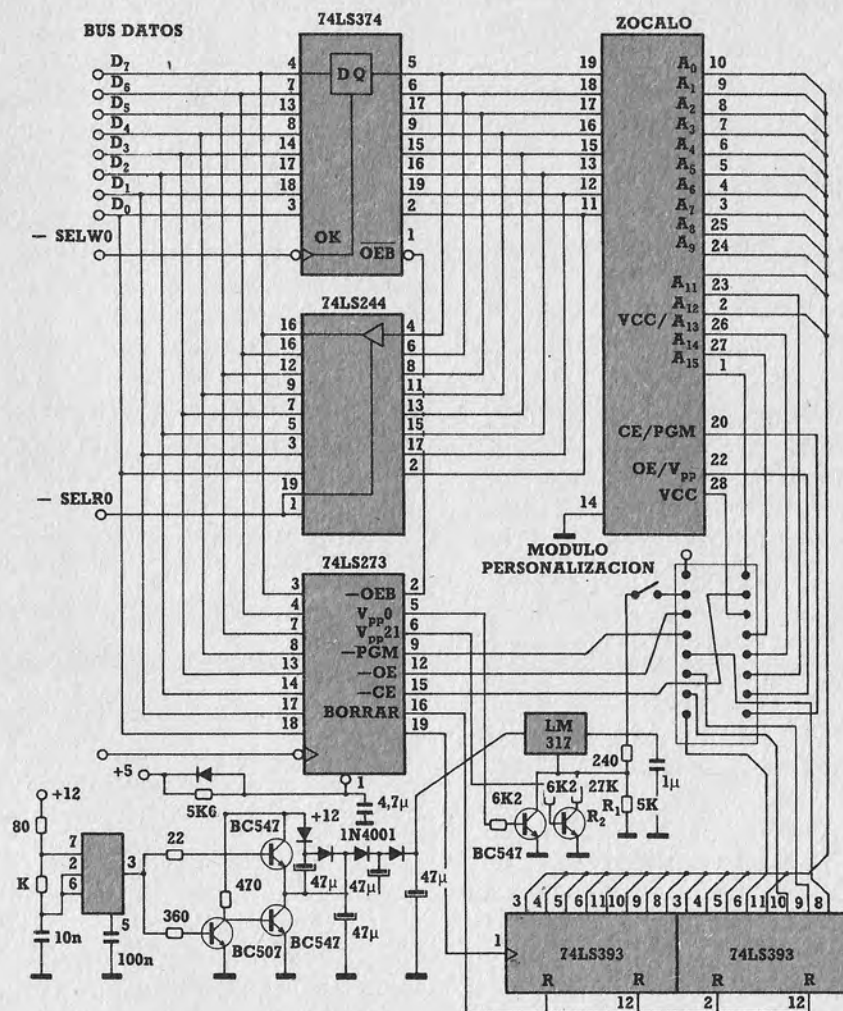


Fig 1. Circuito completo de grabador de EPROMS

EL TALLER DEL HARDWARE

Para una utilización cómoda de la tarjeta pueden colocarse los componentes de acuerdo con el esquema adjunto. El zócalo de la memoria a grabar podría sacarse al exterior mediante cable plano, con objeto de poder trabajar con la máquina cerrada, pues con el zócalo en la tarjeta hay que mantener la tapa quitada para acceder al zócalo.

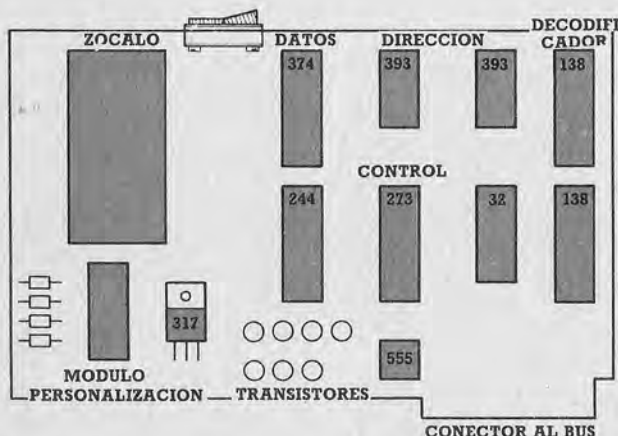


Fig 2. Ubicación de componentes

Programa de grabación

El circuito por si solo no es capaz de realizar las funciones, hay que dotarle de inteligencia. Todos los controles son programables a través de las señales del puerto de control.

El programa adjunto muestra un ejemplo de activación de las funciones de lectura, grabación, acceso a disco, presentación en pantalla y verificación. Puede ampliarse para realizar otras funciones basadas en éstas, para copiar, editar o listar en impresora el contenido de una memoria.

El programa lleva numerosos comentarios sobre las acciones ejecutadas por cada sentencia o grupo de sentencias. Conviene detallar algunas de las más importantes para poder ser modificadas y adaptadas a cada ordenador en particular. Conviene recalcar que es un programa de ejemplo de las posibilidades de la tarjeta de grabación. Es responsabilidad de cada uno hacer que ejecute correctamente las funciones completas.

El programa está dividido en:

- Definición de constantes, variables y direcciones de puertos.

- Menú principal.
- Menú de selección de tipo de EPROM.
- Menú de selección de función a ejecutar.
- Rutinas de funciones particulares.

Las constantes empleadas en el control de puerto de salida se encuentran definidas al principio, junto con un comentario explicativo. Los puertos se encuentran localizados en la zona del adaptador de juegos: X'200'-X'20F'. Para los demás ordenadores deberán usarse las direcciones correspondientes a SELR0, SELW0 y SELW1.

Los menús son muy simples y fáciles de adaptar a cualquier ordenador, pues no se utilizan sentencias LOCATE, que le habrían dado una presentación más atractiva. Son simples PRINT y un número de indicación de la opción. Hay una simple comparación de que la opción seleccionada es válida, para controlar las respuestas. Si se da una opción no válida, un corto pitido indica que se ha de repetir.

El menú de selección de funciones indica las que están disponibles en el ejemplo. Para ampliar las posibilidades es suficiente con añadir la función deseada en una zona libre de identificadores y referenciarla en la sentencia GOSUB que realiza la distribución. Deberá también modificarse el número por el que se determina que una función es admisible, en la sentencia con IF.

Todas las rutinas de interfaz con la memoria se han realizado en BASIC excepto la de temporización para la grabación. La rutina se ha escrito en lenguaje de máquina y tiene la siguiente secuencia de instrucciones:

```
52 PUSH DX GUARDA LOS REGISTROS A EMPLEAR
50 PUSH AX
51 PUSH CX
BA 01 02 MOV DX,X'201' DIRECCIONA PUERTO
B9 B0 61 MOV CX,X'600B' BUCLE DE APROX 50 MILISEC
B0 2B MOV AL,X'29' ACTIVA PULSO PGM
EE OUT DX,AL
E2 FE L1 LOOP L1 DECREMENTA CX HASTA CX=0
B0 3B MOV AL,X'39' DESACTIVA PGM
EE OUT DX,AL
59 POP CX RESTAURA REGISTROS
58 POP AX
5A POP DX
CB RET VUELVE AL PROGRAMA
```

El programa se carga en la zona de memoria del BASIC mediante POKes en el espacio de una variable del tipo vector de enteros. La constante que se carga en CX fija la duración del período de grabación, junto con la ve-

locidad del reloj de la máquina. Es por tanto necesario tener en cuenta el modelo exacto de PC y cambiar esta constante para que el pulso sea de 50 milisegundos, lo más exactos que se pueda.

La ejecución del programa puede resultar un poco lenta en general, pero como es una necesidad poco frecuente el grabar una EPROM no parece importante.

Si se desea mayores prestaciones puede hacerse el programa en lenguaje de máquina completamente o utilizarse un lenguaje compilable.

```

10 REM *****
20 REM PROGRAMA DE GRABACION DE EPROMS
30 REM *****
40 REM UTILIZA LA TARJETA DE GRABACION UNIVER-
SAL
50 REM EN EL IBM PC
60 REM *****
61 OPTION BASE 0
62 DEFINT A-Z
63 DIM RUTINA(40)
64 DIM MEMORIA(16384)
65 REM DESCRIPCION DEL PUERTO DE CONTROL:
66 REM BIT 7 = -OEB, ACTIVA LAS SALIDAS DEL REGIS-
TRO DATOS
67 REM BIT 6 = Vpp0, ACTIVA TENSION DE 0 VOLTIOS
68 REM BIT 5 = Vpp21, ACTIVA LA TENSION DE GRA-
BACION
69 REM BIT 4 = -PGM, PULSOS LOGICOS DE GRABA-
CION
70 REM BIT 3 = -OE, PERMITE LA LECTURA DE LA
EPROM
71 REM BIT 2 = -CE, SELECCION DE LA EPROM
72 REM BIT 1 = BORRADO DE CONTADOR DE DIREC-
CION
73 REM BIT 0 = RELOJ, AVANZA CONTADOR DE DIREC-
CION
74 REM ***** DEFINICION DE ESTADOS DE CONTROL
*****
75 REPOSO=&H9C: REM NIVELES DE REPOSO AL
ARRANCAR
76 BORRAR=&H2: REM PULSO DE BORRADO DE CON-
TADORES
77 CONTAR=&H1: REM PULSO DE AVANZAR DIREC-
CION, INVERTIDO
78 PGM=&H10: REM PULSO DE GRABACION, NIVEL LO-
GICO
79 ELECTURA=&H91: REM ESTADO DE LECTURA DE
EPROM
80 EVERIFICA=&HB1: REM ESTADO DE VERIFICACION
DE EPROM
81 VPP21=&H20: REM NIVEL DE GRABACION ACTIVA-
DO
82 VPP0=&H40: REM NIVEL 0 DE PROGRAMACION
83 INCREMENTAL=&H90: REM LECTURA CON PULSO
DE INCREMENTO
84 INCREMENTAG=&H38: REM GRABACION CON PUL-
SO DE INCREMENTO
85 EGRABACION=&H39: REM ESTADO DE GRABACION
DE EPROM
90 REM ***** DEFINICION DE ESCALARES

```

```

100 OPCION=1: NMAX=2047: TIPOEPROM=2716
106 REM ***** DEFINICION DE PUERTOS
107 PDATE=&H201: REM DATOS
108 PCONTROL=&H203: REM BYTE DE CONTROL
109 REM ***** NUMERO DE BYTES DE CADA EPROM
110 DATA 2047,4095,8191,16383,32767
111 FOR I=1 TO 5: READ Q(I): NEXT I
112 DATA 2716,2732,2764,27128,27256
113 FOR I=1 TO 5: READ TIPO(I): NEXT I
114 DATA &H52,&H50,&H51,&HBA,&H01,
&H02,&HB9,&HB0
115 DATA &H61,&HB0,&H2B,&HEE,&HE2,
&HFE,&HB0,&H3B
116 DATA &HEE,&H59,&H58,&H5A,&HCB
117 TIEMPO50=VARPTR(RUTINA(0)): REM RUTINA DE
50 MILISEGS
118 FOR I=0 TO 20
119 READ VA
120 POKE TIEMPO50+I,VA
121 NEXT I
125 GOSUB 3000: REM PONE TARJETA EN REPOSO
128 KEY OFF: TEPROM=1
129 CLS: REM ***** MENU PRINCIPAL
130 TIPOEPROM=TIPO(TEPROM)
131 PRINT " PROGRAMA DE GRABACION DE EPROMS "
140 PRINT
142 PRINT " VERIFIQUE QUE EL MODULO DE PERSONA-
LIZACION"
143 PRINT " COINCIDE CON EL TIPO DE EPROM A UTILI-
ZAR"
147 PRINT
148 PRINT " TIPO DE EPROM SELECCIONADA=",TIPOE-
PROM
149 PRINT
150 PRINT " SELECCIONE OPCION:"
160 PRINT
170 PRINT " 1.-SELECCIONAR TIPO DE EPROM"
190 PRINT
200 PRINT " 2.-SELECCIONAR FUNCION"
210 PRINT
220 PRINT " 3.-TERMINAR"
260 PRINT
270 INPUT "OPCION=";OPCION
280 IF OPCION<1 OR OPCION>3 THEN BEEP: GOTO
129
290 ON OPCION GOSUB 1000,2000,3000
295 IF OPCION=3 THEN END
300 GOTO 129
990 REM ***** MENU DE TIPOS DE EPROM *****
1000 CLS: REM SELECCION DE TIPO DE EPROM
1010 PRINT " MENU DE SELECCION DE TIPO DE EPROM"
1020 PRINT
1030 PRINT "1.-2716 ( DE 5 VOLTIOS )
1040 PRINT
1050 PRINT "2.-2732"
1060 PRINT
1070 PRINT "3.-2764"
1080 PRINT
1090 PRINT "4.-27128"
1100 PRINT
1110 PRINT "5.-27256"
1140 PRINT
1150 INPUT "TIPO DE EPROM=";TEPROM
1160 IF TEPROM<1 OR TEPROM>5 THEN BEEP:GOTO
1000
1170 REM ESTABLECE PARAMETROS DE GRABACION
1200 NMAX=Q(TEPROM)
1500 RETURN

```

EL TALLER DEL HARDWARE

```

1990 REM ***** MENU DE FUNCIONES *****
2000 CLS: REM FUNCIONES CON LA EPROM
2010 PRINT "SELECCION DE FUNCIONES"
2020 PRINT
2030 PRINT "1.-LEER EPROM A MEMORIA"
2040 PRINT
2050 PRINT "2.-GRABAR CONTENIDO DE MEMORIA A EPROM"
2060 PRINT
2070 PRINT "3.-SALVAR CONTENIDO DE MEMORIA EN DISCO"
2080 PRINT
2090 PRINT "4.-LEER DE DISCO A MEMORIA"
2100 PRINT
2110 PRINT "5.-VERIFICAR QUE LA EPROM ESTA BORRADA"
2120 PRINT
2130 PRINT "6.-COMPARAR EPROM CON MEMORIA"
2140 PRINT
2141 PRINT "7.-PRESENTAR LISTADO DE EPROM"
2142 PRINT
2145 PRINT "8.-PRESENTAR LISTADO DE MEMORIA"
2146 PRINT
2150 PRINT "9.-VOLVER AL MENU PRINCIPAL"
2160 PRINT
2170 INPUT "FUNCION=";FUNCION
2180 IF FUNCION<1 OR FUNCION>9 THEN BEEP: GOTO 2000
2190      ON      FUNCION      GOSUB
4000,5000,6000,7000,8000,9000,
10000,11000,12000
2200 IF FUNCION=9 THEN RETURN ELSE GOTO 2000
2990 REM ***** FUNCION DE PUESTA EN REPOSO *****
3000 REM DEJA LA TARJETA EN REPOSO
3010 OUT PCONTROL,REPOSO
3100 RETURN
3990 REM ***** FUNCION DE LECTURA DE EPROM ***
*****

4000 REM LEE EPROM A MEMORIA
4002 PRINT "ATENCION, PONGA EL CONMUTADOR EN POSICION LEER"
4004 GOSUB 15000
4006 PRINT
4008 PRINT "LEYENDO EPROM A MEMORIA"
4009 ESTADO=ELECTURA OR BORRAR
4010 OUT PCONTROL,ESTADO: REM PONE CONTADOR A CERO
4011 ESTADO=ESTADO XOR BORRAR
4015 OUT PCONTROL,ESTADO: REM PONE MODO LECTURA
4020 FOR I=0 TO NMAX
4040 MEMORIA(I)=INP(PDATO)
4045 OUT PCONTROL,INCREMENTAL
4046 OUT PCONTROL,ELECTURA
4050 NEXT I
4850 GOSUB 15000
4900 RETURN
4990 REM ***** FUNCION DE GRABACION DE EPROM *****

5000 REM GRABA EPROM DESDE MEMORIA
5002 PRINT "ATENCION, PONGA EL CONMUTADOR EN POSICION GRABAR"
5004 GOSUB 15000
5006 PRINT
5008 PRINT "GRABANDO EPROM DESDE MEMORIA"
5009 ESTADO=EGRABACION OR BORRAR
5010 OUT PCONTROL,ESTADO: REM PONE CONTADOR A CERO

5011 ESTADO=ESTADO XOR BORRAR
5015 OUT PCONTROL,ESTADO: REM PONE MODO GRABACION
5020 FOR I=0 TO NMAX
5040 OUT DATO,MEMORIA(I)
5042 CALL TIEMPO50
5045 OUT PCONTROL,INCREMENTAG
5047 OUT PCONTROL,EGRABACION
5050 NEXT I
5990 REM ***** FUNCION DE GRABACION A DISCO ***
*****

6000 REM GRABA DATOS DE MEMORIA A DISCO
6002 INPUT "NOMBRE DEL FICHERO CON DATOS";NOMF2#
6004 OPEN NOMF2# FOR OUTPUT AS #2
6006 PRINT
6008 PRINT "GRABANDO DATOS EN DISCO"
6020 FOR I=0 TO NMAX
6040 PRINT #2,MEMORIA(I)
6050 NEXT I
6100 CLOSE #2
6850 GOSUB 15000
6900 RETURN
6990 REM ***** FUNCION DE LEER DE DISCO *****
7000 REM LEER DATOS DE DISCO A MEMORIA
7002 INPUT "NOMBRE DEL FICHERO CON DATOS";NOMF1#
7004 OPEN NOMF1# FOR INPUT AS #1
7006 PRINT
7008 PRINT "LEYENDO DATOS DE DISCO"
7020 FOR I=0 TO NMAX
7040 INPUT #1,MEMORIA(I)
7050 NEXT I
7100 CLOSE #1
7850 GOSUB 15000
7900 RETURN
7990 REM ***** FUNCION DE VERIFICACION DE EPROM *****

8000 REM VERIFICA QUE LA EPROM ESTA BORRADA
8006 PRINT
8009 ESTADO=ELECTURA OR BORRAR
8010 OUT PCONTROL,ESTADO: REM PONE CONTADOR A CERO
8011 ESTADO=ESTADO XOR BORRAR
8015 OUT PCONTROL,ESTADO: REM PONE MODO LECTURA
8017 PRINT "VERIFICANDO EPROM BORRADA"
8020 FOR I=0 TO NMAX
8040 XA=INP(PDATO)
8045 IF XA=255 THEN 8065
8060 PRINT "POSICION=";HEX$(I);" EPROM=";HEX$(XA)
8065 OUT PCONTROL,INCREMENTAL
8067 OUT PCONTROL,ELECTURA
8070 NEXT I
8850 GOSUB 15000
8900 RETURN
8990 REM ***** FUNCION DE COMPARACION DE EPROM Y RAM *****

9000 REM COMPARA MEMORIA CON EPROM
9009 ESTADO=ELECTURA OR BORRAR
9010 OUT PCONTROL,ESTADO: REM PONE CONTADOR A CERO
9011 ESTADO=ESTADO XOR BORRAR
9015 OUT PCONTROL,ESTADO: REM PONE MODO LECTURA
9017 PRINT "COMPARANDO MEMORIA CON EPROM"
9020 FOR I=0 TO NMAX
9040 XA=INP(PDATO)
9050 IF XA=MEMORIA(I) THEN 9070

```

```

9060 PRINT "POSICION=";HEX$(I); " EPROM=";HEX$(XA);
9065 PRINT " MEMORIA=";MEMORIA(I)
9070 OUT PCONTROL,INCREMENTAL
9080 OUT PCONTROL,ELECTURA
9100 NEXT I
9200 GOSUB 15000
9300 RETURN
9990 REM * * * * * FUNCION DE PRESENTACION DE
EPROM * * * * *
10000 REM LEE EPROM A PANTALLA
10010 OUT PCONTROL,ELECTURA OR BORRAR: REM
PONE CONTADOR A CERO
10015 OUT PCONTROL,ELECTURA: REM PONE MODO
LECTURA
10017 PRINT "LISTADO DEL CONTENIDO DE LA EPROM"
10020 FOR I=0 TO NMAX
10040 XA=INP(PDATO)
10050 PRINT HEX$(XA); " ";
10060 OUT PCONTROL,INCREMENTAL
10070 OUT PCONTROL,ELECTURA
10200 NEXT I
10490 GOSUB 15000
10500 RETURN
10990 REM * * * * * FUNCION DE PRESENTACION DE ME-
MORIA * * * * *
11000 REM PRESENTA MEMORIA A PANTALLA
11010 PRINT "LISTADO DEL CONTENIDO DE LA MEMO-
RIA"
11020 FOR I=0 TO NMAX
11050 PRINT HEX$(MEMORIA(I)); " ";
11200 NEXT I
11490 GOSUB 15000
11500 RETURN
12000 RETURN
15000 REM * * * * * PETICION DE PULSAR TECLA
15010 PRINT
15020 INPUT "PULSE UNA TECLA PARA CONTI-
NUAR";X$
15100 RETURN

```

Las EPROM más recientes permiten ser grabadas mediante un ciclo un poco más complejo pero que reduce en gran medida el tiempo de grabación. El método denominado "grabación inteligente" consiste en alimentar con tensión ligeramente superior y verifica después de cada pulso de grabación el estado en el que queda la posición. Si está ya grabada, se le dan unos pocos pulsos más y se pasa a la celda siguiente, si no lo está se le da un pulso más y se repite el ciclo hasta un número determinado de veces. Requiere una circuitería un poco más compleja, pero reduce el tiempo de programación a una quinta parte, por lo que es el método empleado en los programadores industriales.

Los EPROMS 27128, 27256 y 27512 utilizan tensión de grabación de 12,5v. Es necesario ajustar R1 y R2 para estas tensiones.

■ Borrador de EPROMS

Para el borrado de las EPROM es necesaria una fuente de luz ultravioleta de longi-

tud de onda inferior a 5000 Å. La construcción es sencilla a partir de un pequeño tubo fluorescente de los utilizados como germicida y al cual no se le ha pintado la capa interior para que genere luz blanca como efecto secundario. Pueden encontrarse de 6, 8, 12 y 24 vatios. Cualquiera de ellos puede servir para esta aplicación.

El circuito para su funcionamiento es el mismo que para un tubo fluorescente corriente: necesitamos una reactancia y un cebador. Para un funcionamiento cómodo y seguro se necesita también una caja, un conmutador y es recomendable un temporizador para regular el tiempo de borrado. El tiempo normal para borrado de las EPROMS oscila entre 15 y 30 minutos, colocadas a una distancia de 10 a 20 milímetros del tubo.

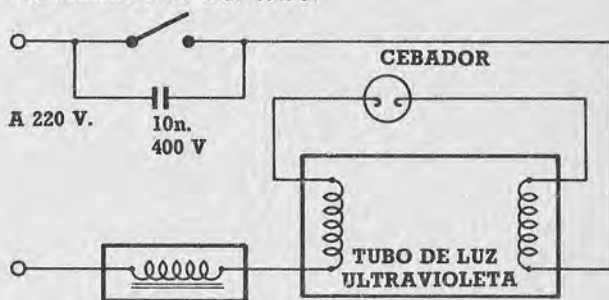


Fig 3. Circuito eléctrico del borrador de EPROMs.

Es necesario recalcar que la luz ultravioleta que emite el tubo no es visible pero puede dañar muy seriamente a la vista. La luz violeta, que sí se ve, es un indicio de que hay emisión ultravioleta también. La caja debería disponer de un microconmutador conectado a la tapa para impedir que pudiera funcionar sin estar cerrada completamente.

Las memorias a borrar deberán colocarse a una distancia de un centímetro aproximadamente, con la ventana próxima al tubo.

El esquema de la figura indica una posible solución práctica de borrador de EPROMS.

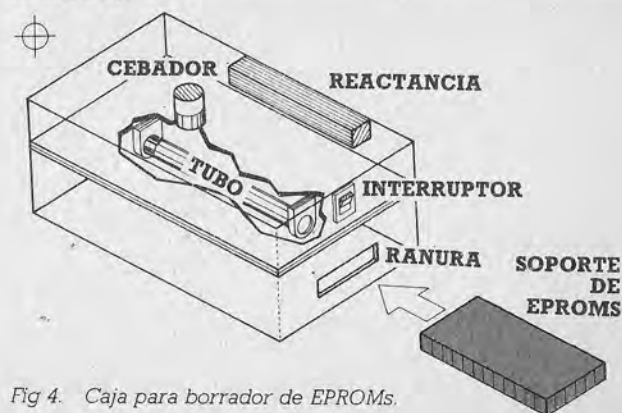
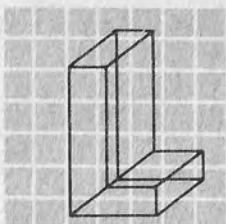


Fig 4. Caja para borrador de EPROMs.

MATEMATICAS: Vectores



OS vectores son muy utilizados para representar magnitudes físicas. Ejemplos muy claros de ello los tenemos en la velocidad y la fuerza.

Para definir la velocidad de un automóvil, hay que dar su valor numérico, la dirección que lleva y el sentido, ya que el coche no tardará lo mismo si va por una carretera o por otra, ni si va más rápido o más lento.

Lo mismo nos ocurre con la fuerza, ya que el objeto empujado se moverá más o menos según desde dónde lo empujemos y hacia dónde. Es importante que aprendas a manejar los vectores y a operar con ellos.

Para empezar te presentamos un programa que realiza operaciones básicas; tú seleccionas en el menú principal la que deseas hacer en cada ocasión e introduces los datos que te pida el ordenador.

1. **MODULO.** Es la longitud del vector (Fig. 1).

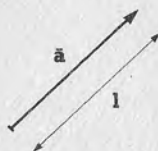


Fig. 1

2. **SUMA DE VECTORES.** Es el vector que tiene como origen el origen del primer vector y como extremo el extremo del segundo cuando están puestos a continuación (Figura 2).

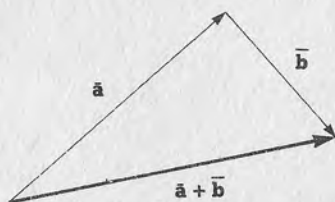


Fig. 2

3. **DIFERENCIA DE VECTORES.** Se suma al primer vector el opuesto del segundo (Fig. 3).

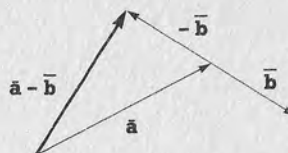


Fig. 3

4. **PRODUCTO POR UN ESCALAR.** Es el vector que se obtiene al poner, tantas veces como marque el escalar, el vector sobre una recta (Fig. 4).



Fig. 4

5. **PRODUCTO ESCALAR.** El producto escalar de dos vectores es el resultado de multiplicar el módulo de ambos por el coseno del ángulo que forman.

6. **VECTORES EQUIPOLENTES.** Dos vectores son equipolentes si el cuadrilátero obtenido de unir sus extremos es un paralelogramo (Fig. 5).

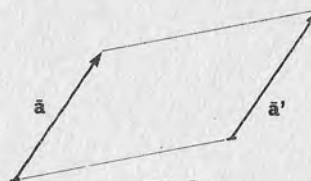


Fig. 5

Cada vez que realices una de las operaciones puedes volver al menú principal con sólo pulsar una tecla.

```

10 REM *****
20 REM ** CALCULOS VECTORIALES **
30 REM *****
40 CLS
50 REM *****
60 REM ** MENU DE OPCIONES **
70 REM *****
80 PRINT "      1.- MODULO DE UN VECTOR"
90 PRINT
100 PRINT "      2.- SUMA DE VECTORES"
110 PRINT
120 PRINT "      3.- DIFERENCIA DE DOS VECTORES"
130 PRINT
140 PRINT "      4.- PRODUCTO POR UN ESCALAR"
150 PRINT
160 PRINT "      5.- PRODUCTO ESCALAR"
170 PRINT
180 PRINT "      6.- EQUIPOLENCIA"
190 PRINT
200 PRINT
210 INPUT "¿QUE OPCION DESEAS?":R
220 REM *****
230 REM ** BIFURCACIONES DEL PROGRAMA **
240 REM *****
250 IF R=1 THEN GOSUB 400
260 IF R=2 THEN GOSUB 500
270 IF R=3 THEN GOSUB 600
280 IF R=4 THEN GOSUB 700
290 IF R=5 THEN GOSUB 800
300 IF R=6 THEN GOSUB 900
310 GOTO 40
320 END
400 REM *****
410 REM ** SUBROUTINA MODULO **
420 REM *****
430 CLS
440 INPUT "COORDENADA X DEL VECTOR :":X
450 INPUT "COORDENADA Y DEL VECTOR :":Y
460 LET M=X^2:LET N=Y^2:LET M=M+N:LET M=SQR(M)
470 PRINT "EL MODULO DEL VECTOR  ("X:",Y:")  ES :M
480 INPUT W: RETURN
500 REM *****
510 REM ** SUBROUTINA SUMA **
520 REM *****
530 CLS
540 INPUT "COORDENADA X DEL VECTOR 1 :":X1
550 INPUT "COORDENADA Y DEL VECTOR 1 :":Y1
560 INPUT "COORDENADA X DEL VECTOR 2 :":X2
570 INPUT "COORDENADA Y DEL VECTOR 2 :":Y2
580 PRINT "EL VECTOR SUMA DE  ("X1:",Y1:")  Y  ("X2:",Y2:")  ES  ("X1
+X2:",Y1+Y2:")"
590 INPUT W: RETURN
600 REM *****
610 REM ** SUBROUTINA RESTA **
620 REM *****
630 CLS
640 INPUT "COORDENADA X DEL VECTOR 1 :":X1
650 INPUT "COORDENADA Y DEL VECTOR 1 :":Y1
660 INPUT "COORDENADA X DEL VECTOR 2 :":X2
670 INPUT "COORDENADA Y DEL VECTOR 2 :":Y2
680 PRINT "EL VECTOR RESTA DE  ("X1:",Y1:")  Y  ("X2:",Y2:")  ES  ("X
1-X2:",Y1-Y2:")"
690 INPUT W: RETURN
700 REM *****
710 REM ** SUBROUTINA PRODUCTO POR UN ESCALAR **
720 REM *****
730 CLS
740 INPUT "COORDENADA X DEL VECTOR :":X
750 INPUT "COORDENADA Y DEL VECTOR :":Y
760 INPUT "ESCALAR :":A
770 PRINT "EL VECTOR RESULTANTE DE  ("X:",Y:")  POR EL ESCALAR "A:" ES  ("
XA:",YA:")"
780 INPUT W: RETURN
800 REM *****
810 REM ** SUBROUTINA PRODUCTO ESCALAR **
820 REM *****
830 CLS
840 INPUT "COORDENADA X DEL VECTOR 1 :":X1
850 INPUT "COORDENADA Y DEL VECTOR 1 :":Y1
860 INPUT "COORDENADA X DEL VECTOR 2 :":X2
870 INPUT "COORDENADA Y DEL VECTOR 2 :":Y2

```

```

880 PRINT "EL PRODUCTO ESCALAR DE ("X1:", "Y1:") Y ("X2:", "Y2:") ES
":X1*X2+Y1*Y2
890 INPUT W: RETURN
900 REM *****
910 REM ** SUBROUTINA EQUIPOLENCIA **
920 REM *****
930 CLS
940 INPUT "COORDENADA X DEL VECTOR 1 :":X1
950 INPUT "COORDENADA Y DEL VECTOR 1 :":Y1
960 INPUT "COORDENADA X DEL VECTOR 2 :":X2
970 INPUT "COORDENADA Y DEL VECTOR 2 :":Y2
980 LET R$="SON EQUIPOLENTES"
990 IF (X1/X2)<>(Y1/Y2) THEN LET R$="NO "+R$
1000 PRINT "LOS VECTORES ("X1:", "Y1:") Y ("X2:", "Y2:") ":R$
1010 INPUT W: RETURN

```

VARIACIONES PARA OTROS ORDENADORES

ZX-SPECTRUM

320 GOTO 9999

COMMODORE

```

40 PRINT CHR$(147)
430 PRINT CHR$(147)
530 PRINT CHR$(147)
630 PRINT CHR$(147)
730 PRINT CHR$(147)
830 PRINT CHR$(147)
930 PRINT CHR$(147)

```

NATURALEZA Y TECNOLOGIA

Química: formulación química

La formulación química es una de las cosas más necesarias de aprender, sobre todo para aquellos que soñáis ser grandes científicos, inventores y descubridores de cosas nuevas.

Por otra parte, también es cierto que a veces es algo difícil llegar a conocerla bien: la gran cantidad de compuestos existentes y

de métodos de formulación son los principales motivos.

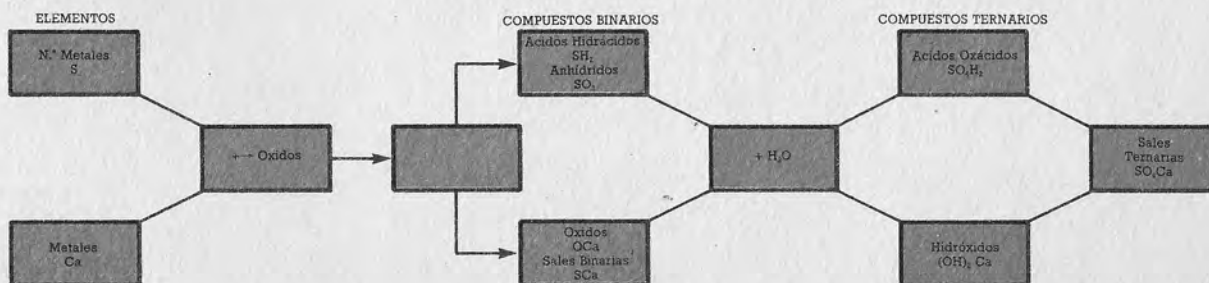
Vamos a presentaros aquí dos programas de formulación, el primero actúa como alumno resolviendo vuestras preguntas, formulará los compuestos que le propongáis por el método sistemático.

El segundo programa actúa como un profesor, os hará un test para ver cómo anda vuestra formulación; para ello ha utilizado el método tradicional.

No os desaniméis, con su ayuda y vuestro esfuerzo llegaréis a formular como grandes químicos.

	Elementos	Valencia con el hidrógeno	Valencia con el oxígeno
	Hidrógeno	—	1
Halógenos	Ftlor	1	—
	Cloro	1	1, 3, 5, 7
	Bromo	1	1, 3, 5, 7
	Iodo	1	1, 3, 5, 7
Anfígenos	Oxígeno	2	—
	Azufre	2	2, 4, 6
Nitrogenoides	Nitrógeno	3	1, 3, 5
	Fósforo	3	1, 3, 5
	Arsenico	3	1, 3, 5
	Antimonio	3	1, 3, 5
Carbonoides	Carbono	4	2, 4
	Silicio	4	2, 4

Alcalinos	Litio	1
	Sodio	1
	Potasio	1
Alcalino-térreos	Calcio	2
	Estroncio	2
	Bario	2
	Radio	2
Térreos	Boro	3
	Aluminio	3
Metales de transición	Cobre	1 y 2
	Plata	1
	Oro	1 y 3
	Magnesio	2
	Cinc	2
	Mercurio	1 y 2
	Hierro	2 y 3
	Níquel	2 y 3
	Cobalto	2 y 3
	Platino	4
	Iridio	4
	Cromo	3 y 6
	Manganeso	4 y 7
	Pomo	2 y 4
	Estaño	2 y 4



APRENDER CON EL ORDENADOR

```

10 REM *****
20 REM ** FORMULACION INORGANICA POR EL METODO SISTEMATICO **
30 REM *****
40 CLS:LET T=0
50 REM *****
60 REM ** CREACION DE TABLAS DE ELEMENTOS **
70 REM *****
80 LET T1=80:LET T2=18:LET T3=14:LET T4=14
90 DIM T1$(T1),T2$(T2),T3$(T3),T4$(T4)
100 FOR I=1 TO T1
110 READ T1$(I)
120 NEXT I
130 FOR I=1 TO T2
140 READ T2$(I)
150 NEXT I
160 FOR I=1 TO T3
170 READ T3$(I)
180 NEXT I
190 FOR I=1 TO T4
200 READ T4$(I)
210 NEXT I
220 REM *****
230 REM ** LECTURA DE LA FORMULA **
240 REM *****
250 PRINT " FORMULACION INORGANICA POR EL METODO SISTEMATICO"
260 PRINT :PRINT
270 PRINT " Recuerda que debes introducir la formula con la primera letra d
el
compuesto en mayusculas v las demas en minusculas."
280 PRINT :PRINT
290 INPUT "¿ES UN COMPUESTO BINARIO O TERNARIO? (B/T)":B$
300 PRINT :PRINT
310 INPUT "INTRODUCE LA FORMULA ":F$
320 LET N=LEN(F$)
330 REM *****
340 REM ** IDENTIFICACION DE ELEMENTO Y SUBINDICE **
350 REM *****
360 DIM A$(N),A(N)
370 FOR I=1 TO N
380 LET A$(I)=MID$(F$,I,1)
390 LET A=ASC(A$(I))
400 IF (A-48)>0 AND (A-48)<10 THEN LET A(I)=1
410 NEXT I
420 IF B$="T" THEN GOTO 700
430 REM *****
440 REM ** FORMULACION DEL COMPUESTO BINARIO **
450 REM *****
460 FOR I=N TO 1 STEP-1
470 LET S=ASC(A$(I))
480 IF A(I)=1 THEN LET Q$=A$(I):GOSUB 1000: GOTO 530
490 IF A(I-1)=1 THEN LET Q$=A$(I):GOSUB 1200: GOTO 530
500 IF A(I-1)=0 AND S>91 THEN LET Q$=A$(I-1)+A$(I):LET I=I-1:GOSUB 1200: GOTO 53
0
510 LET Q$=A$(I)
520 GOSUB 1200
530 NEXT I
540 PRINT R$
550 END
700 REM *****
710 REM ** FORMULACION DEL COMPUESTO TERNARIO **
720 REM *****
730 FOR I=N TO 1 STEP-1
740 LET S=ASC(A$(I))
750 IF A(I)=1 THEN LET Q$=A$(I):GOSUB 1000: GOTO 800
760 IF A(I-1)=1 THEN LET Q$=A$(I):GOSUB 1400: GOTO 800
770 IF A(I-1)=0 AND S>91 THEN LET Q$=A$(I-1)+A$(I):LET I=I-1:GOSUB 1400: GOTO 80
780 LET Q$=A$(I)
790 GOSUB 1400
800 NEXT I
810 GOTO 540
1000 REM *****
1010 REM ** SUBROUTINA DE SUBINDICES **
1020 REM *****
1030 FOR J=1 TO T4 STEP 2
1040 IF Q$=T4$(J) THEN LET R$=R$+T4$(J+1):GOTO 1060
1050 NEXT J
1060 RETURN
1200 REM *****
1210 REM ** SUBROUTINA DE COMPUESTOS BINARIOS **
1220 REM *****
1230 LET T=T+1
1240 IF T>1 THEN GOTO 1280

```

```

1250 FOR J=1 TO T2 STEP 2
1260 IF Q$=T2$(J) THEN LET R$=R$+T2$(J+1)+" de ":GOTO 1310
1270 NEXT J
1280 FOR J=1 TO T1 STEP 2
1290 IF Q$=T1$(J) THEN LET R$=R$+T1$(J+1):GOTO 1310
1300 NEXT J
1310 RETURN
1400 REM *****
1410 REM ** SUBROUTINA DE COMPUESTOS TERNARIOS **
1420 REM *****
1430 LET T=T+1
1440 IF T=3 THEN GOTO 1500
1450 IF T=2 THEN GOTO 1540
1460 FOR J=1 TO T3 STEP 2
1470 IF Q$=T3$(J) THEN LET R$=R$+T3$(J+1):GOTO 1570
1480 NEXT J
1490 GOTO 1570
1500 FOR J=1 TO T1 STEP 2
1510 IF Q$=T1$(J) THEN LET R$=R$+T1$(J+1):GOTO 1570
1520 NEXT J
1530 GOTO 1570
1540 FOR J=1 TO T3 STEP 2
1550 IF Q$=T3$(J) THEN LET R$=R$+T3$(J+1)+" de ":GOTO 1570
1560 NEXT J
1570 RETURN
5000 DATA H.Hidrogeno,Li,Litio,Na,Sodio,K,Potasio,Rb,Rubideo,Cs,Cesio,Fr,Francio
5010 DATA Be,Berilio,Mg,Magnesio,Ca,Calcio,Sr,Estroncio,Ba,Bario,Ra,Radio
5020 DATA B,Boro,Al,Aluminio
5030 DATA C,Carbono,Si,Silicio
5040 DATA N,Nitrogeno,As,Arsenico,Sb,Antimonio,Bi,Bismuto
5050 DATA O,Oxigeno,S,Azufre,Se,Selenio,Te,Teluro
5060 DATA F,Fluor,Cl,Cloro,Br,Bromo,I,Iodo
5070 DATA Cu,Cobre,Ag,Plata,Au,Oro,Mn,Manganeso,Zn,Cinc,Hg,Mercurio,Fe,Hierro,Pt
,Platino,Cr,Cromo,Pb,Plomo,Sn,Estaño
6000 DATA D,Oxido,H,Hidruro
6010 DATA F,Fluoruro,Cl,Cloruro,Br,Bromuro,I,Ioduro,S,Sulfuro,Se,Seleniuro,Te,Tel
ururo
7000 DATA O,Oxo,S,Sulfato,Se,Seleniato
7010 DATA Cl,Clorato,Br,Bromato,P,Fosfato,I,Iodato
8000 DATA 1,Mono,2,Di,3,Tri,4,Tetra,5,Penta,6,Hexa,7,Hepta

```

VARIACIONES PARA OTROS ORDENADORES

ZX-SPECTRUM

```

90 DIM T1$(T1,10),T2$(T2,10),
T3$(T3,10),T4$(T4,10)
550 GOTO 9999

```

Las sentencias DATA deben llevar los
datos entre comillas por ser alfanuméricos.

COMMODORE

```

40 PRINT CHR$(147):LET T=0

```

```

10 REM *****
20 REM ** FORMULACION QUIMICA POR EL METODO TRADICIONAL **
30 REM *****
40 CLS
50 REM *****
60 REM ** NUMERO DE DATOS Y PREGUNTAS **
70 REM *****
80 PRINT "FORMULACION QUIMICA POR EL METODO TRADICIONAL"
90 PRINT:PRINT:PRINT
100 PRINT "...PARA REALIZAR LA FORMULACION DEBE PONER EN MAYUSCULAS LAS INICIALE
S DE LAS PALABRAS Y DE LOS COMPUESTOS"
110 LET N=38:LET PREG=10
120 REM *****
130 REM ** LECTURA DE MATRIZ DE DATOS **
140 REM *****
150 DIM M$(2*N)
160 FOR I=1 TO 2*N
170 READ A$:LET M$(I)=A$
180 NEXT I
190 REM *****
200 REM ** PREGUNTAS DEL TEST **
210 REM *****
220 FOR K=1 TO PREG
230 LET Q=INT(RND*2*N)+1

```

```

240 GOSUB 450
250 IF A =1 THEN GOTO 230
260 NEXT K
270 REM *****
280 REM ** RESPUESTAS TEST **
290 REM *****
300 FOR I = 1 TO PREG
310 PRINT I: "-":M$(T(I))
320 IF T(I)=2*INT(T(I)/2) THEN INPUT "Introduzca formula...":F$
330 IF T(I)<>2*INT(T(I)/2) THEN GOTO 370
340 IF F$ =M$(T(I)-1) THEN GOTO 410
350 PRINT M$(T(I)-1)
360 GOTO 420
370 INPUT "Introduzca el nombre del compuesto...":W$
380 IF W$ =M$(T(I)+1) THEN GOTO 410
390 PRINT M$(T(I)+1)
400 GOTO 420
410 PRINT "!!!CORRECTO !!!":GOTO 430
420 PRINT "MAL "
430 NEXT I
440 END
450 REM *****
460 REM ** SUBROUTINA COMPUESTOS **
470 REM *****
480 LET T(K)=0
490 FOR P=1 TO K-1
500 IF Q=T(P) THEN LET A=1: GOTO 530
510 NEXT P
520 LET A=0
530 RETURN
1000 DATA "H2SO4","Acido Sulfurico","HCl","Acido Clorhidrico","NaCl","Cloruro So
dico"
1010 DATA "K2CrO4","Cromato de Potasio","Ag2S2O3","Tiosulfato de Plata","(NH4)2S
O5","Peroxosulfato de Amonio"
1020 DATA "Mg(IO2)2","Iodito de Magnesio","NaAsO3","Metaarseniato de Sodio","Fe2
P2O7","Difosfato de Hierro(IV)"
1030 DATA "Al(ClO)3","Hipoclorito de Aluminio","Ag2CrO4","Cromato de Plata","NaH
MnO4","Hidrogenomanganato de Sodio"
1040 DATA "NH4CN","Cianuro de Amonio","NH4SCN","Tiocianato de Amonio","SrMnO4","
Manganato de Estroncio"
1050 DATA "H2O2","Peroxido de Hidrogeno","Ag4P2O7","Pirofosfato de Arsenico","Sr
CO3","Carbonato de Estroncio"
1060 DATA "Mg(ClO4)2","Perclorato de Magnesio","MnS","Sulfuro de Manganeso(II)
1070 DATA "LiHS","Hidrogenosulfuro de Litio","NaHCO3","Bicarbonato de Potasio","
Fe(OH)2","Hidroxido Ferroso","Ba(BrO)2","Hipobromito de Bario"
1080 DATA "Li4C","Carburo de Litio","K3SbO4","Antimoniato de Potasio","AgCN","Ci
anuro de Plata","CCl4","Tetracloruro de Carbono"
1090 DATA "H2Se","Acido Selenhidrico","H4Sb2O5","Acido Piroantimonioso","BH3","B
orano"
1100 DATA "CH4","Metano","SiH4","Silano","NH3","Amoniaco","PH3","Fosfina","AsH3
","Arsina","SbH3","Estibina","H2O","Agua"

```

VARIACIONES PARA OTROS ORDENADORES

ZX-SPECTRUM

```

150 DIM M$(2*N,23)
440 GOTO 9999

```

COMMODORE

```

40 PRINT CHR$(147)

```

SOCIEDAD

Geografía: Los volcanes

Probablemente, es el vulcanismo el proceso magmático mejor conocido por el hombre. Esto se debe a que una parte, al menos, del proceso puede ser observada con relativa

facilidad. No obstante, el vulcanismo no es aún totalmente conocido.

Recordemos, en primer lugar, qué es un volcán.

Un volcán puede definirse como una fisura o grieta de la corteza terrestre que pone en comunicación a las masas magmáticas internas con la superficie.

El magma es una mezcla de materiales fundidos (principalmente silicatos), gases en disolución (dióxido de carbono, flúor, cloro...) y productos sólidos (no fundidos o ya cristalizados a partir de las moléculas dispersas del propio magma).

Una característica importante del magma es su viscosidad; en ella influye, entre otros factores: la composición química, la temperatura, la presión... Al mismo tiempo, es esta

característica la que más va a influir en el tiempo de erupción volcánica; de éstas se distinguen cuatro tipos que surgieron al producirse erupciones reales y, por ello, llevan nombres de volcanes conocidos: Hawaiano, Peleano y Estromboliano.

Es posible que estos fenómenos te resulten enigmáticos y en ocasiones te hayas hecho preguntas como:

¿De dónde proceden los magmas?

Se supone que proceden de zonas diversas de la Tierra denominadas subcorticales, si se originan por debajo de la corteza terrestre, e intracorticales, si se originan en zonas continentales a poca profundidad.

¿Cómo se originan?

En cualquiera de las zonas se debe a la energía térmica existente, los materiales se funden, dando lugar a los magmas.

¿Cómo ascienden?

Se debe principalmente a diferencias de presión, viscosidad y densidad entre el magma y la roca encajante.

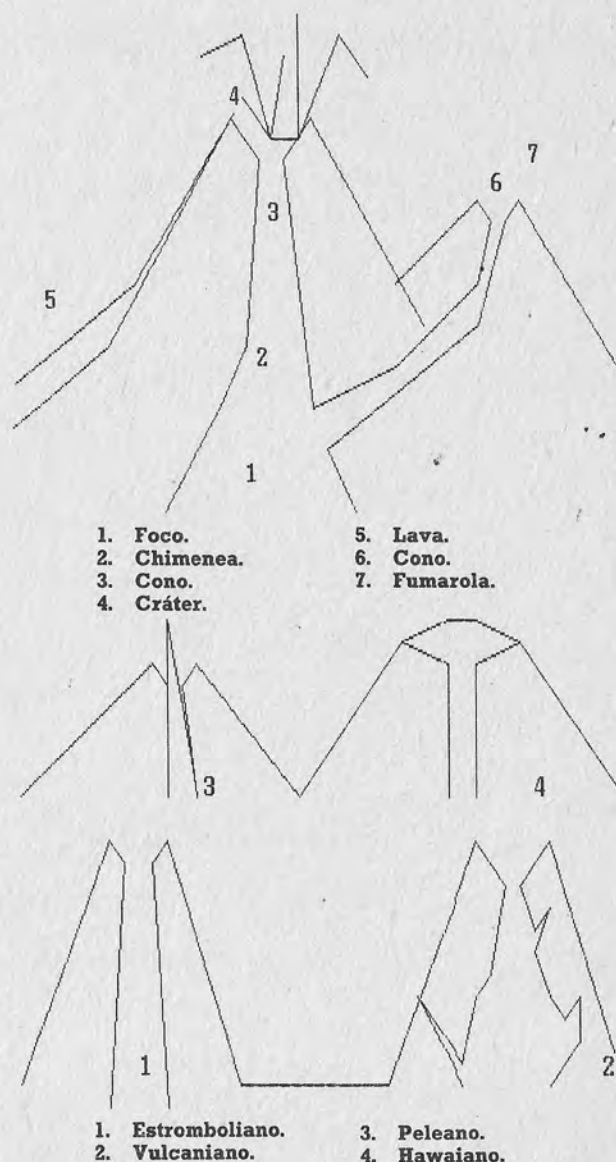
En el camino ascendente el magma se va enfriando y solidificándose.

La fuerza que lleva le hace salir al exterior, produciéndose así la erupción volcánica.

Una vez recordados algunos puntos, puedes fijar tus conocimientos con la ayuda del ordenador; el programa que te presentamos consta de dos opciones:

— Diagramas de las partes de un volcán y sus tipos.

— Test de conocimientos.



```

10 SCREEN 2
20 CLS
30 PRINT " V      V  OOO  L      CCCC  AAA  N      N  EEEE  SSS  "
40 PRINT " V      V  O   O L      C   A   A  N N  N  E   S   "
50 PRINT " V      V  O   O L      C   AAAAA N  N  N  EEE  SSSS "
60 PRINT " V V   O   O L      C   A   A  N  N N E   S"
70 PRINT " V      OOO  LLLL  CCCC  A   A  N      N  EEEE  SSSS
80 PRINT
90 PRINT
100 PRINT
110 PRINT "
120 PRINT "
130 PRINT "
140 PRINT "
150 PRINT "
160 PRINT "
170 PRINT "
180 PRINT
190 PRINT
200 INPUT "¿QUE OPCION DESEAS":R
210 IF R=1 THEN GOSUB 500:GOTO 20
220 IF R=2 THEN GOSUB 4000:GOTO 20
230 END
500 REM *****
510 REM ** PROGRAMA DIBUJOS VOLCAN **

```

```

520 REM *****
530 CLS
540 RESTORE 3030
550 LET Y=300-50
560 PSET (2,Y)
570 FOR I=1 TO 6
580 READ A,B
590 LET B=B*(-10):LET B$=STR$(B):LET A=A*10:LET A$=STR$(A)
600 LET DIB$="M"+" "+A$+" "," "+"B$
610 DRAW DIB$
620 NEXT I
630 LET Y=300-10
640 PSET (250,Y)
650 FOR I=1 TO 5
660 READ A,B
670 LET B=B*(-10):LET B$=STR$(B):LET A=A*10:LET A$=STR$(A)
680 LET DIB$="M"+" "+A$+" "," "+"B$
690 DRAW DIB$
700 NEXT I
710 LET Y=300-100
720 PSET (300,Y)
730 FOR I=1 TO 8
740 READ A,B
750 LET B=B*(-10):LET B$=STR$(B):LET A=A*10:LET A$=STR$(A)
760 LET DIB$="M"+" "+A$+" "," "+"B$
770 DRAW DIB$
780 NEXT I
790 LET Y=300-70
800 PSET (1,Y)
810 FOR I=1 TO 14
820 READ A,B
830 LET B=B*(-10):LET B$=STR$(B):LET A=A*10:LET A$=STR$(A)
840 LET DIB$="M"+" "+A$+" "," "+"B$
850 DRAW DIB$
860 NEXT I
870 GOSUB 1500
880 INPUT 0
1000 REM *****
1010 REM ** DIBUJO TIPOS DE VOLCANES **
1020 REM *****
1030 CLS
1040 LET Y=300-20
1050 PSET (1,Y)
1060 FOR I=1 TO 3
1070 READ A,B
1080 LET B=B*(-10):LET B$=STR$(B):LET A=A*10:LET A$=STR$(A)
1090 LET DIB$="M"+" "+A$+" "," "+"B$
1100 DRAW DIB$
1110 NEXT I
1120 LET Y=300-10
1130 PSET (100,Y)
1140 FOR I=1 TO 11
1150 READ A,B
1160 LET B=B*(-10):LET B$=STR$(B):LET A=A*10:LET A$=STR$(A)
1170 LET DIB$="M"+" "+A$+" "," "+"B$
1180 DRAW DIB$
1190 NEXT I
1200 LET Y=300-20
1210 PSET (360,Y)
1220 FOR I=1 TO 10
1230 READ A,B
1240 LET B=B*(-10):LET B$=STR$(B):LET A=A*10:LET A$=STR$(A)
1250 LET DIB$="M"+" "+A$+" "," "+"B$
1260 DRAW DIB$
1270 NEXT I
1280 LET Y=300-150
1290 PSET (1,Y)
1300 FOR I=1 TO 20
1310 READ A,B
1320 LET B=B*(-10):LET B$=STR$(B):LET A=A*10:LET A$=STR$(A)
1330 LET DIB$="M"+" "+A$+" "," "+"B$
1340 DRAW DIB$
1350 NEXT I
1360 GOSUB 2000:RETURN
1500 REM *****
1510 REM ** NOMBRES **
1520 REM *****
1530 LOCATE 20,20:PRINT "1"
1540 LOCATE 16,21:PRINT "2"
1550 LOCATE 11,22:PRINT "3"
1560 LOCATE 7,19:PRINT "4"
    
```

```

1570 LOCATE 14,4:PRINT "5"
1580 LOCATE 10,40:PRINT "6"
1590 LOCATE 9,38:PRINT "7"
1600 FOR I=1 TO 7
1610 READ N$
1620 LOCATE 1,1:PRINT I;"-":N$
1630 INPUT "PRESIONA UNA TECLA PARA CONTINUAR ":A
1640 NEXT I
1650 RETURN
2000 REM *****
2010 REM ** TIPOS **
2020 REM *****
2030 LOCATE 20,10:PRINT "1"
2040 LOCATE 20,45:PRINT "2"
2050 LOCATE 11,15:PRINT "3"
2060 LOCATE 11,40:PRINT "4"
2070 FOR I=1 TO 4
2080 READ N$
2090 LOCATE 1,1:PRINT I;"-":N$
2100 NEXT I
2110 INPUT A
2120 RETURN
3000 REM *****
3010 REM ** DATOS **
3020 REM *****
3030 DATA +7,+4,+9,+11,+2,-2,-1,-9,-2,-3,-4,-5
3040 DATA -2,+3,+11,+6,+2,+5,+1,+1,+7,-8
3050 DATA -8,+10,-2,-2,+2,-12,+6,+2,+6,+4,+1,+3,-1,+1,-6,-4
3060 DATA +9,+5,+8,+9,+2,-2,+5,-3,-1,+3,+1,+2,-5,+1,+4,-1,-4,+2,+0,+0,+6,+0,-
6,+3,+5,+2,-2
3070 DATA "FOCO", "CHIMENEA", "CONO", "CRATER",
", "LAVA", "CONO ADVENTICIO", "FUMAROLA",
3080 DATA +6,+11,+1,-1,-1,-11
3090 DATA -1,+11,+1,+1,+5,-11,+10,+0,+6,+11,+2,-2,-1,-4,-1,-1,-1,-3,-3,+3,+3,-4
3100 DATA +2,+2,+0,+2,-1,-1,-1,+1,-1,+2,+1,+2,-1,-1,-1,+2,+2,+5,-11
3110 DATA +9,+6,+1,-1,+0,-5,+0,+8,+2,-8,-1,+5,+1,+1,+7,-6,+7,+7,+3,-1,+0,-6,+0,+
6,-3,+1,+3,+1,+2,+0,+3,-1,+7,-7,+7,-3,-1,+0,-6
3120 DATA "ESTROMBOLIAND", "VULCANIAND", "PELEAND", "HAWAIAND"
3130 RETURN
4000 REM *****
4010 REM ** PROGRAMA PREGUNTAS VOLCAN **
4020 REM *****
4030 CLS
4040 RESTORE 5000
4050 LET P=16:LET PREG=16
4060 DIM P$(P*2)
4070 FOR I=1 TO P*2
4080 READ P$(I)
4090 NEXT I
4100 FOR I=1 TO PREG
4110 PRINT "ESCRIBE LA RESPUESTA EN MAYUSCULAS":PRINT
4120 PRINT P$(I*2-1)
4130 INPUT R$
4140 IF R$=P$(I*2) THEN GOTO 4170
4150 PRINT "LA RESPUESTA ES ":P$(I*2)
4160 GOTO 4180
4170 PRINT "!!!CORRECTO!!!":FOR Z=1 TO 500:NEXT Z
4180 NEXT I
4190 RETURN
5000 DATA "El conjunto de rocas fundidas, mezcladas con los gases desprendidos r
ecibe el nombre de ","MAGMA"
5010 DATA "Cuando el magma asciende por una grieta se forman los ","VOLCANES"
5020 DATA "El nucleo candente o bolsa de magma recibe el nombre de ","FOCO"
5030 DATA "La grieta principal que comunica el foco con el exterior es la
","CHIMENEA"
5040 DATA "La parte en forma de cono recibe el nombre de ","CONO"
5050 DATA "La cavidad de la parte alta se denomina ","CRATER"
5060 DATA "Cuando un volcan está en actividad se dice que está en ","ERUPCION"
5070 DATA "Las erupciones de magma, poco violentas y sin explosiones son de tipo
","HAWAIAND"
5080 DATA "Los volcanes MAUNA, LOA Y KILAUEA son de tipo ","HAWAIAND"
5090 DATA "Las erupciones de magma fluido que arrojan productos solidos son del
tipo ","ESTROMBOLIAND"
5100 DATA "El volcan Estromboli es de tipo ","ESTROMBOLIAND"
5110 DATA "Las erupciones de lava espesa que se solidifica con rapidez, son del
tipo ","VULCANIAND"
5120 DATA "Los volcanes VESUBIO y VULCANO son de tipo ","VULCANIAND"
5130 DATA "Cuando la lava es tan espesa que se solidifica sobre el crater se dic
e que es del tipo ","PELEAND"
5140 DATA "El volcan del MONT PELE es del tipo ","PELEAND"
5150 DATA "Las emanaciones gaseosas que se escapan por las grietas se llaman ","
FUMAROLAS"

```

APRENDER CON EL ORDENADOR

VARIACIONES PARA OTROS ORDENADORES

ZX-SPECTRUM

```
10 no se pone      1130 PLOT 100,Y
230 GOTO 9999      1160 no se pone
550 LET Y=50       1170 no se pone
560 PLOT 2,Y       1180 DRAW A*5,B*5
590 no se pone     1200 LET Y=20
600 no se pone     1210 PLOT 360,Y
610 DRAW A*5,B*5   1240 no se pone
630 LET Y=10       1250 no se pone
640 PLOT 250,Y     1260 DRAW A*5,B*5
670 no se pone     1280 LET Y=150
680 no se pone     1290 PLOT 1,Y
690 DRAW A*5,B*5   1320 no se pone
710 LET Y=100      1330 no se pone
720 PLOT 300,Y     1340 DRAW A*5,B*5
750 no se pone     1530 PRINT AT 20,20;"1"
760 no se pone     1530 PRINT AT 16,21;"2"
770 DRAW A*5,B*5   1530 PRINT AT 11,22;"3"
790 LET Y=70       1530 PRINT AT 7,19;"4"
800 PLOT 1,Y       1530 PRINT AT 14,4;"5"
830 no se pone     1530 PRINT AT 10,40;"6"
840 no se pone     1530 PRINT AT 9,30;"7"
850 DRAW A*5,B*5   1620 PRINT AT 1,1;I;"-",N$
1040 LET Y=20      2030 PRINT AT 20,10;"1"
1050 PLOT 1,Y      2040 PRINT AT 20,45;"2"
1080 no se pone    2050 PRINT AT 11,15;"3"
1090 no se pone    2060 PRINT AT 11,40;"4"
1100 DRAW A*5,B*5  2090 PRINT AT 1,1;I;"-",N$
1120 LET Y=10      4060 DIM P$(P*2,60)
```

AMSTRAD

```
10 MODE 2          680 no se pone
550 LET Y=50        690 DRAW A*10,B*10
560 PLOT 2,Y        710 LET Y=100
590 no se pone      720 PLOT 300,Y
600 no se pone      750 no se pone
610 DRAW A*10,B*10  760 no se pone
630 LET Y=10        770 DRAW A*10,B*10
640 PLOT 250,Y      790 LET Y=70
670 no se pone      800 PLOT 1,Y
```

```
830 no se pone
840 no se pone
850 DRAW A*10,B*10
1040 LET Y=20
1050 PLOT 1,Y
1080 no se pone
1090 no se pone
1100 DRAW A*10,B*10
1120 LET Y=10
1130 PLOT 100,Y
1160 no se pone
1170 no se pone
1180 DRAW A*10,B*10
1200 LET Y=20
1210 PLOT 360,Y
1240 no se pone
1250 no se pone
1260 DRAW A*10,B*10
1280 LET Y=150
1290 PLOT 1,Y
1320 no se pone
1330 no se pone
1340 DRAW A*10,B*10
1540 LOCATE 21,16:PRINT "2"
1550 LOCATE 22,11:PRINT "3"
1560 LOCATE 7,19:PRINT "4"
1570 LOCATE 4,14:PRINT "5"
1580 LOCATE 40,10:PRINT "6"
1590 LOCATE 88,9:PRINT "7"
2030 LOCATE 10,20:PRINT "1"
2040 LOCATE 45,20:PRINT "2"
2050 LOCATE 15,11:PRINT "3"
2060 LOCATE 40,11:PRINT "4"
2090 LOCATE 1,1:PRINT I;"-",N$
```

PARA LOS MAS JOVENES

■ Circulación vial

Tú, aunque seas pequeño, también tienes derecho a tu examen de circulación, podrás obtener tu carné de conducir cuando seas mayor, pero no por ello debes dejar de conocer ahora algunas normas básicas.

Para empezar, debes ser capaz de superar un test que te proponemos en el programa; contiene los conceptos más elementales que debes saber.

```
10 REM *****
20 REM ** PROGRAMA SOBRE LA CIRCULACION VIAL **
30 REM *****
40 CLS
50 LET C=0
60 REM *****
70 REM ** LECTURA DE PREGUNTAS DEL TEST **
80 REM *****
```

```

90 DIM P$(40): DIM R(10)
100 FOR I=1 TO 40
110 READ P$(I)
120 NEXT I
130 REM *****
140 REM ** LECTURA DE RESPUESTAS DEL TEST **
150 REM *****
160 FOR I=1 TO 10
170 READ R(I)
180 NEXT I
190 REM *****
200 REM ** ENCABEZAMIENTO CON CONTADOR Y COCHE **
210 REM *****
220 PRINT " 0 1 2 3 4 5 6 7 8 9 10 META"
230 PRINT " ### # #"
240 PRINT "##### # #"
250 PRINT " 0 0 # #"
260 REM *****
270 REM ** PREGUNTAS TEST **
280 REM *****
290 FOR I=1 TO 10
300 LOCATE 10,1:PRINT P$(I*4-3)
310 LOCATE 12,5:PRINT P$(I*4-2)
320 LOCATE 14,5:PRINT P$(I*4-1)
330 LOCATE 16,5:PRINT P$(I*4)
340 PRINT
350 INPUT N
360 CLS
370 GOSUB 1530
380 IF N=R(I) THEN LET C=C+1
390 GOSUB 1030
400 NEXT I
410 IF C=10 THEN GOSUB 2000
420 END
1000 REM *****
1010 REM ** SUBROUTINA CONTADOR **
1020 REM *****
1030 LOCATE 2,C*5-4
1040 PRINT " ### "
1050 LOCATE 3,C*5-4
1060 PRINT " #####"
1070 LOCATE 4,C*5-4
1080 PRINT " 0 0 "
1090 RETURN
1500 REM *****
1510 REM ** SUBROUTINA CAMINO **
1520 REM *****
1530 PRINT " 0 1 2 3 4 5 6 7 8 9 10 META"
1540 PRINT " # #"
1550 PRINT " # #"
1560 PRINT " # #"
1570 RETURN
2000 REM *****
2010 REM ** GANADOR **
2020 REM *****
2030 PRINT :PRINT :PRINT :PRINT
2040 PRINT " *****"
2050 PRINT " * *"
2060 PRINT " * *"
2070 PRINT " * *"
2080 PRINT " ***"
2090 PRINT " ***"
2100 PRINT " ***"
2110 PRINT " *****"
2120 PRINT
2130 PRINT " CAMPEON "
2140 RETURN
3000 DATA "EN LA CIUDAD LOS COCHES CIRCULAN POR LA :","1-LA ACERA","2-LA CALZADA",
"3-LAS CALLES PEATONALES"
3010 DATA "YO DEBO CRUZAR CUANDO EL SEMAFORO DE PEATONES ESTE:","1-EN VERDE","2-EN AMARILLO","3-EN ROJO"
3020 DATA "LOS ENCARGADOS DE PONER ORDEN EN EL TRAFICO SON:","1-LOS BOMBEROS","2-LOS GUARDIAS DE TRAFICO","3-LOS POLICIAS NACIONALES"
3030 DATA "LOS COCHES LOS CONDUCEN:","1-LOS MAYORES","2-LOS NIÑOS","3-LAS NIÑAS"
3040 DATA "LOS VEHICULOS DE TRANSPORTE PUBLICO SON:","1-LAS MOTOS","2- LOS COCHE S","3-LOS AUTOBUSES"
3050 DATA "LOS NIÑOS CUANDO VAN EN COCHE DEBEN SENTARSE EN:","1-EL ASIENTO TRASE RO","2-EL MALETERO","3-EL ASIENTO DELANTERO"
3060 DATA "LOS AUTOMOVILES PARA EL TRANSPORTE DE MERCANCIAS SE LLAMAN:","1-MOTOS","2-AUTOBUSES","3-CAMIONES"

```

APRENDER CON EL ORDENADOR

```
3070 DATA "LOS AUTOBUSE RECOGEN A LAS PERSONAS QUE QUIEREN VIAJAR EN ELLOS:", "1-  
EN LAS BOCAS DE METRO", "2-EN CUALQUIER SITIO", "3-EN LAS PARADAS DE AUTOBUS"  
3080 DATA "LOS COCHES DEBEN PARAR CUANDO:", "1-EL SEMAFORO ESTA VERDE", "2-NO HAY  
SEMAFORO", "3-HAY UN PASO DE CEBRA"  
3090 DATA "LA GENTE DEBE CONDUCIR:", "1-A TODA VELOCIDAD", "2-CON PRUDENCIA", "3-MU  
Y MUY DESPACIO"  
3100 DATA 2,1,2,1,3,1,3,3,3,2
```

MODIFICACIONES PARA OTROS ORDENADORES

ZX-SPECTRUM

```
90 DIM P$(40,50):DIM R(10)  
300 PRINT AT 10,1;P$(I*4-3)  
310 PRINT AT 12,5;P$(I*4-2)  
320 PRINT AT 14,5;P$(I*4-1)  
330 PRINT AT 16,5;P$(I*4)  
440 GOTO 9999  
1030 PRINT AT 2,C*5-4;"   ###"  
1040 no se pone  
1050 PRINT AT 3,C*5-4;"   #####"  
1060 no se pone  
1070 PRINT AT 4,C*5-4;"   0  0"
```

AMSTRAD

```
300 LOCATE 1,10: PRINT P$(I*4-3)  
310 LOCATE 5,12: PRINT P$(I*4-2)
```

```
320 LOCATE 5,14: PRINT P$(I*4-1)  
330 LOCATE 5,16: PRINT P$(I*4)  
1030 LOCATE C*5-4,2  
1050 LOCATE C*5-4,3  
1070 LOCATE C*5-4,4
```

MSX

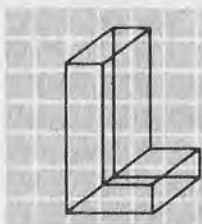
No hay variaciones

COMMODORE

```
40 PRINT CHR$(147)  
360 PRINT CHR$(147)
```

En las líneas 300, 310, 320, 330, 1030, 1050, 1070 debes utilizar la rutina que se daba en el primer fascículo para situar el cursor en una posición de la pantalla.

PEQUEÑA HISTORIA DE LA INFORMATICA



A empresa Apple no existía hace diez años. Sus fundadores, Stephen Jobs y Steve Wozniak, estaban trabajando en dos empresas del valle, Atari y Hewlett Packard. La historia de Stephen Jobs

no parece muy alegre ni fácil. Es un niño abandonado, adoptado por Clara y Paul Jobs. Su carácter no es muy abierto, y es una persona meditabunda. Como buen contestatario, no acepta en principio la comida tradicional, se hace vegetariano, y prueba la meditación y el LSD. Por fin, comienza a trabajar en Atari, una pequeña empresa del valle, muy innovadora. Ha diseñado un ping pong electrónico para jugar utilizando el ordenador. En Atari tampoco gana demasiados amigos, ya que resulta muy solitario y excéntrico. Como la sociedad es muy pequeña y el trato muy personalizado, la dirección llega a un acuerdo con Jobs para que trabaje por la noche. Pero Jobs no se ha adaptado, sigue haciéndose preguntas que no tienen respuesta. Abandona Atari, y parte hacia la India, buscando un sentido a su vida. A su vuelta, no ha logrado tampoco avanzar mucho en su contacto con las demás personas. Intenta localizar a sus verdaderos padres, pero tampoco lo consigue. Finalmente, este inadaptado se integra en un club informático, y es en él donde conoce al que será su socio y amigo: Steve Wozniak. Wozniak es un hombre de negocios: para él no basta con crear algo innovador, hay que darlo a conocer, venderlo, y obtener el dinero necesario para seguir trabajando. Como los dos hombres son muy jóvenes, sin contactos ni experiencia empresarial, Wozniak introduce en el negocio al mejor comercial de Intel, Mike Makkula.

En el verano de 1976, el prototipo del nuevo ordenador Apple, el Apple I, aparece en proveedores especializados, que les encargan varios pedidos.

¿Por qué el nombre de Apple? Sencillamente, porque Jobs había engordado mucho, y estaba obsesionado en adelgazar comiendo manzanas. Y como tenían verdadera prisa en adoptar un buen nombre, y ninguno se ocupaba seriamente de este pequeño problema, se pusieron un tope de tiempo, pasado el cual si no tenían un nombre a gusto de todos, pondrían al ordenador sencillamente Apple. Ese nombre puede parecer una tontería, pero ha resultado ser un gran acierto. En primer lugar, es diferente a cualquier otro de los seleccionados para los ordenadores, y, en segundo lugar, tiene cierto encanto de juventud, de innovación.

Tomando previsiones para el futuro, y también con fines de difusión comercial, la empresa instaura la norma de regalar un cierto número de máquinas a las Universidades. Esa acción favorece muchísimo su difusión. Como el precio de las máquinas es relativamente asequible (desde luego está absolutamente al alcance de pequeñas empresas), la acción es un acierto total de marketing.

Desde el año 1976, la empresa ha subido «algunos» escalones. Hoy, casi la cuarta parte del mercado de microordenadores es de Apple.

Otra firma, Vector, también es fruto de las cavilaciones de dos jóvenes. En este caso se trata del matrimonio Harp. Bob, es ingeniero y creó un prototipo de ordenador. Su mujer se sintió suficientemente valerosa como para comercializarlo. Y la empresa sigue creciendo saludablemente, a pesar de que el matrimonio se separó.

En el valle del Silicio es fácil encontrar titulados en cualquier disciplina, muy especializados. Basta ir a alguna de las fiestas locales. Por ello, es rentable invertir en investigación en la zona. Los investigadores se encuentran ya establecidos, y las tecnologías más nuevas están al alcance de la mano incluso (los técnicos se van informando sobre los productos ya

PEQUEÑA HISTORIA DE LA INFORMATICA

desarrollados sin casi darse cuenta en reuniones de amigos, en charlas con otros ingenieros de la zona, etc.). No estamos hablando de «secretos industriales», sólo de la tecnología disponible.

El gobierno americano tiene asignados muchos proyectos, de todo tipo a empresas de la zona. No es extraño, California es uno de los Estados más industrializados y avanzados. El Departamento de Defensa es el que más invierte en tecnología, siendo el capital considerable (indirectamente, casi un tercio de todos los recursos dedicados a investigación). La palabra investigación no es muy explícita. Puede englobar muchas facetas. Se puede, por ejemplo, investigar sobre las lejanas galaxias, y se puede investigar sobre mercados, sobre las industrias competitivas, sobre la incidencia de esta segunda revolución industrial, sobre el individuo, etc. En el valle de Santa Clara existen empresas dedicadas a investigar sobre galaxias (como Ames, al frente de la cual está un antiguo colaborador de la NASA, el profesor Billingham), naturalmente con un proyecto financiado por el gobierno americano, también, los hay dedicadas a investigar sobre los efectos de la industrialización en la sociedad (profesor Yudken, de la Universidad de Stanford).

El valle del Silicio supone una reacción contra el sistema IBM. Esta empresa es gigante, cuida a sus empleados, les ofrece un futuro seguro, con unas retribuciones bastante buenas, etc., a cambio de lealtad y disposición. El valle del Silicio, por el contrario, está lleno de pequeñas empresas creadas por individuos que estaban dispuestos a perder su estatus cómodo, para pasar a trabajar en garajes y buhardillas, sin ningún tipo de seguridad para su futuro. Es la contraposición. (A pesar de ello, IBM es propietario del 12 % del capital de INTEL, y tiene opciones para obtener hasta el 30 % del capital.)

El futuro del valle, sin embargo, tiene

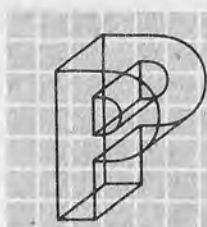
claroscuros. Los precios se han ido abaratando hasta unos puntos en los que el espectacular crecimiento se ha ido reduciendo. Además, y aunque muchos empresarios consideren que «son los elegidos de Dios», la verdad es que existen muchos otros que temen cada vez más al poderío japonés. Entre ellos, el más concienciado es Gordon Bell, vicepresidente de Digital. En la Conferencia de la Quinta Generación de Tokio (1981) participaron varios ingenieros de la empresa. Vinieron muy preocupados por el futuro de la industria informática americana, si no se toman a tiempo las medidas adecuadas.

El partido republicano de los Estados Unidos, algunos de cuyos miembros prominentes tienen intereses en el valle del Silicio, apoya plenamente la idea de prepararse frente al proyecto japonés de Quinta Generación.

Otro aspecto a considerar, en el camino de dar la réplica a los japoneses en sus magníficos planes para la década siguiente, es el de, por un lado, unificar las investigaciones, para no malgastar esfuerzos, y por otro, evitar el espionaje industrial, sobre todo a nivel internacional. El primer punto puede llevarse a cabo creando un organismo gubernamental que controle y sirva de catalizador en las investigaciones. El segundo punto es difícil de solucionar. (Realmente, el valle del Silicio en estos últimos años podía ser una réplica moderna de la Viena de posguerra. El tráfico de secretos industriales y los escándalos están a la orden del día.)

Además, los japoneses han invertido mucho en California. En ese Estado están establecidas más de 700 empresas japonesas. Aunque al principio estas empresas intentaban emular a IBM, en los últimos tiempos tienen potencia suficiente para fabricar productos nuevos y originales. Son el segundo país mundial en industria informática, y, desde luego, el competidor más serio del valle del Silicio.

PROGRAMACION ORIENTADA AL OBJETO (II)



ARA concluir esta somera presentación de los lenguajes orientados al objeto, veamos brevemente las ventajas y desventajas que aparecen con este tipo de programación.

■ Ventajas

Los lenguajes orientados al objeto tienen muchas ventajas sobre los lenguajes tradicionales orientados al procedimiento. La ocultación de la información y la abstracción de datos incrementan la fiabilidad y ayudan a separar la representación de la implementación que se realice. La ligazón dinámica incrementa la flexibilidad permitiendo añadir nuevas clases de objetos (tipos de datos) sin tener que modificar el código existente. La herencia, añadida a la ligazón dinámica, permite volver a utilizar el código. Esto último tiene la ventaja de reducir el volumen del código fuente que hay que programar, incrementando la productividad del programador, ya que necesitará escribir menos código original. La herencia mejora la «distribución del lenguaje fuente» («code factoring»). «Distribución de lenguaje fuente» significa que el código que realiza una tarea específica sólo se encuentra en un único lugar, y esta característica facilita la tarea del mantenimiento del software.

■ Desventajas

Los lenguajes orientados al objeto tienen unas cuantas características que pueden ser consideradas por algunos como desventajas. La más debatida es el coste en tiempo

de ejecución del mecanismo de ligazón dinámica. El envío de un mensaje necesita más tiempo que una llamada de función que se haga directamente. Algunos estudios han mostrado que con un gestor de mensajes bien implementado, esta particularidad supone aproximadamente 1,75 veces el tiempo requerido para una llamada a una función estándar. Las diferencias actuales en la velocidad de ejecución entre los lenguajes tradicionales y sus contrapartidas orientadas al objeto no parecen ser significativas. Esto se debe probablemente al hecho de que el overhead (carga adicional de trabajo) sólo es aplicable a envíos de mensajes y que estos envíos de mensajes llevan a cabo más de una llamada de función. Con frecuencia, una parte del trabajo que se realiza automáticamente con un envío de mensaje, debe hacerlo el programador en cualquier caso, utilizando llamadas de función adicionales o incluso llamadas de función múltiples. De hecho, puede suceder a veces que, en aplicaciones muy grandes, la habilidad de estandarizar y ajustar la funcionalidad proporcionada con los envíos de mensaje puede hacer que la aplicación funcione más rápidamente que su contrapartida tradicional. La razón principal es que el envío de mensajes obvia en gran medida la variabilidad que resulta de los distintos estilos de programación y niveles de habilidad. El envío de mensajes también elimina el código complejo, necesario con frecuencia cuando los programas tradicionales tienen que simular una ligazón dinámica del tipo de la que proporciona automáticamente un lenguaje orientado al objeto.

Otra de las desventajas que se suele citar con frecuencia es que la implementación de los lenguajes orientados al objeto es más compleja que los lenguajes orientados al procedimiento comparables, ya que el «hueco se-

mántico» entre esos lenguajes y el hardware típico de las máquinas es mayor. Por esta razón, se requiere más simulación de software.

Afortunadamente, el coste de implementación del lenguaje se paga una sola vez para cada máquina.

Otro problema posible es que el programador en lenguajes orientados al objeto debe aprender el contenido y manejo de una librería de clases (normalmente bastante extensa) antes de empezar a poder preparar programas que sean realmente eficientes. Por esto, también, la utilidad en el manejo de los lenguajes orientados al objeto depende enormemente de que se disponga de una buena documentación y de que en la implementación que se esté utilizando se ofrezcan buenas herramientas de desarrollo (menús, ventanas, etcétera) con funciones de manejo de datos y procesos básicos de desarrollo.

Los que comienzan a utilizar los lenguajes orientados al objeto suelen tener dificultades en resolver los problemas de programación desde el punto de vista del objeto.

Normalmente la dificultad estriba en cómo dividir en objetos el nuevo problema de programación. Un buen punto para comenzar es imaginarse que cada fragmento de programa y cada conjunto de datos están flotando juntos en el espacio. Imaginemos a continuación unos trozos de cuerda que unen los fragmentos de código y los datos que utilizan y otros trozos entre los segmentos de código que se utilizan conjuntamente o que realizan una función similar. Sigamos imaginando que movemos el conjunto de elementos que tenemos (unos alrededor de los otros) hasta que la longitud total de la cuerda sea la más corta posible. Rodeando la estructura de cada dato se encontrarán las rutinas que utiliza, y las divisiones naturales del código aparecerán con toda evidencia. La división de un problema en objetos consiste simplemente en colocar las cosas en el lugar donde deben estar.

La división de un determinado problema en objetos y la definición de aquellas acciones que le son «naturales» simplifica enormemente los programas. Si se dividen las acciones de un objeto en las adecuadas porciones, podremos pensar y escribir códigos a un

nivel más alto. Sabemos que los procedimientos a los que llamamos desde el código a nivel alto son correctos porque concuerdan con lo que pensamos sobre el problema. Cuando existe esta seguridad, cometemos menos errores. Tendremos menos preocupación sobre si los procedimientos subordinados están realizando lo que deben realizar.

■ ¿Por qué utilizar lenguajes orientados al objeto?

La parte más importante de la programación orientada al objeto no son las ventajas técnicas que puede proporcionar, sino el hecho de que traspasa el umbral de la percepción. Cuando se coloca toda la información asociada con un nódulo en una instancia de clase, no sólo se simplifica el programa en su conjunto. Nos permite pensar en ese conjunto de información y actuar sobre él como una simple unidad. Percibimos el mundo que nos rodea como formado por objetos, y nuestro cerebro distribuye la información en pedazos. Mediante la utilización de objetos en un lenguaje de programación, podemos profundizar en un determinado patrón de pensamiento existente.

Si pensamos los algoritmos en términos de objetos, nos serán mucho más fáciles de comprender. Esta facilidad de comprensión nace generalmente de los detalles: cómo un procedimiento se ha construido, sin necesidad de pensar sobre el resto del programa. Cuando se trabaja dentro de una clase (en cualquier lenguaje orientado al objeto), se está a salvo de los efectos y complejidades de las otras partes del programa (que están fuera de esa clase).

Un buen diseño y un código limpio no son las únicas ventajas de los lenguajes orientados al objeto. Se pueden escribir códigos muy perfectos en otros lenguajes, y horribles en Smalltalk, por ejemplo.

La programación orientada al objeto permite realizar la mayor parte del trabajo con antelación, almacenarla en una librería y reutilizarla como código genérico en posteriores programas.

VOCABULARIO DE INFORMATICA

EAROM (Electrically-Alterable Read Only Memory). Memoria ROM susceptible de ser borrada por medios eléctricos.

EBCDIC (Extended Binary Coded Decimal Interchange Code). Código binario de 8 bits que deriva del código numérico BCCD.

ECL (Emitter Coupled Logic). Lógica acoplada por emisor. En grandes ordenadores, generalmente tecnología cuya característica más importante es la velocidad.

Eco. Repetición por pantalla de todo lo introducido en el teclado. Muchas instrucciones no producen eco en la pantalla. Por ejemplo, al pulsar instrucciones en un terminal, éstas se envían a la Unidad Central, sin que tengan que aparecer por pantalla. Si aparecen, se habrá producido el «eco». En la mayoría de los casos, la aparición de «eco» es controlable por el usuario.

Editar. Modificar la forma de presentación o formato de los datos, por ejemplo, borrando o insertando caracteres, numeraciones de página, etc.

Editor. Programa que sirve para corregir la edición de textos. Permite insertar partes, borrar, cambiar caracteres o frases completas, etc. Los editores más completos son programas específicos de procesamiento de textos.

EDP (Electronic Data Processing). Proceso Electrónico de Datos. Se refiere a todas aquellas operaciones realizadas con ordenadores electrónicos.

EEPROM (Electrically Erasable Programmable Read Only Memory). Memoria de sólo lectura (ROM) que se puede programar y borrar por medios eléctricos.

Efectiva, dirección. Dirección derivada de aplicar un indexado o direccionamiento indirecto a la dirección especificada, y que se utiliza para identificar la posición real del operando con el que se está trabajando.

Electromagnético, línea de retardo. Línea de retardo cuya operación está basada en el tiempo de propagación de las ondas electromagnéticas, emitadas con una capacitancia e inductancia determinadas.

Electronic data processing. (SC1) Proceso de datos, realizado primordialmente por dispositivos electrónicos. Perteneciente a un equipo de proceso de datos, que es predominantemente electrónico, como, por ejemplo, un ordenador.

Electrostático, almacenamiento. Dispositivo que almacena los datos creando cargas electrostáticas en determinadas zonas de una superficie dieléctrica.

Elemento. Componente individual de un conjunto. (Véase el tipo de elemento, como por ejemplo, **lógico, secuencial, threshold**, etc.).

Emular. Imitar un sistema mediante otro. El sistema imitador aceptará los mismos datos, ejecutará los mismos programas y obtendrá los mismos resultados que el sistema imitado.

Emulador. Dispositivo o programa de ordenador que realiza la acción de emular.

VOCABULARIO DE INFORMATICA

Enmascarar. Sistema por el cual los datos o el programa se procesan según un esquema dado también llamado «máscara».

Ensamblador. Programa (compilador) que se utiliza para traducir otro programa escrito en Ensamblador a lenguaje máquina.

Entero. Número que no tiene una parte decimal. También puede referirse a una variable que representa un entero.

Entrada. Todo dato o texto o programa que se introduce al ordenador para que éste lo procese.

EOF (End of File). Siglas que indican el final de un fichero.

EOT (End of Transmission Character). Carácter que indica el final de una transmisión. También puede referirse al final de una cinta magnética (End of Tape).

EPROM (Erasable Programmable Read Only Memory). Memoria ROM (sólo lectura) susceptible de ser borrada mediante rayos ultravioleta.

Equivalencia. Operador lógico con la siguiente propiedad: si P y Q y R son proposiciones o afirmaciones, la equivalencia entre P, Q y R es verdad si, y sólo si, todas las proposiciones son verdaderas, o si todas ellas son falsas.

Equivalente en dígitos binarios. Número de posiciones binarias que se necesitan para contar los elementos de un conjunto dado.

Error. Diferencia entre una cantidad media u observada, y su valor real, especificado u obtenido teóricamente (véase **Error absoluto**, **Error heredado**, **Error de redondeo**, **Error de truncamiento**).

Error absoluto. Error expresado en las mismas unidades en las que está expresada la unidad que contiene el error. También puede referirse a la magnitud del error sin considerar su signo algebraico.

Error de cuantificación. Error producido en la conversión de una señal analógica en digital. Se debe a que el valor codificado no siempre corresponde exactamente con el valor obtenido en ese instante a partir de la señal analógica.

Error de redondeo. Error producido por una operación de redondeo: Al eliminar las posiciones de menos valor del número, se añade una unidad si el primer dígito eliminado es cinco, o superior, y no se suma nada si dicho primer dígito es cuatro, o menor. En contraposición al error debido a un «truncamiento».

Error de truncamiento. Error debido a una operación de truncado. Es decir, se eliminan los últimos dígitos del número considerado (posiciones menos significativas) decreciendo, por tanto, el valor del número. En contraposición a error de redondeo.

Error de sintaxis. Error producido al haber introducido los datos de forma distinta a como está especificado por las normas de sintaxis correspondientes. Cambio de orden de los parámetros, ausencia de signos de puntuación, paréntesis, etc.

Error heredado. Error producido en un paso anterior (en un proceso secuencial), y acarreado a otros pasos siguientes.

Error, mensaje de. Indicación de que se ha producido un error.

Errores, ráfaga de. En transmisión de datos, secuencia de señales que contienen uno o más errores, que, sin embargo, se cuentan como una unidad, según un determinado criterio.

Errores, código de detección de errores. Código en el que cada expresión es conforme con unas reglas determinadas, de forma que si la expresión resultante no es conforme, inmediatamente se detecta el error. A estos códigos también se les denomina códigos de comprobación automática.

ESC. Se refiere al carácter Escape.

